Accurate alignment of (meta)barcoding data sets using MACSE

Frédéric Delsuc, Vincent Ranwez

Chapter 2.3 Accurate alignment of (meta)barcoding data sets using MACSE

Frédéric Delsuc[1]
Institut des Sciences de l'Evolution de Montpellier (ISEM), CNRS, IRD, EPHE, Université de Montpellier, Montpellier, France
frederic.delsuc@umontpellier.fr
https://orcid.org/0000-0002-6501-6287

Vincent Ranwez[2]
AGAP, Université de Montpellier, CIRAD, INRA, Montpellier SupAgro, Montpellier, France
vincent.ranwez@supagro.fr
https://orcid.org/0000-0002-9308-7541

Abstract

Twenty years of standardized DNA barcoding practice have resulted in millions of sequences being produced for a handful of molecular markers in a wide range of fungi, animal and plant species. Despite some basic quality controls, reference barcoding data sets deposited in the Barcode of Life Datasystem (BOLD) database are not immune to sequencing errors and undetected pseudogenes. Such database inaccuracies can significantly bias subsequent species delimitation and biodiversity estimation based on DNA barcoding. These potential problems are amplified in metabarcoding studies containing thousands of sequences produced using high throughput sequencing technologies. Here, we propose a pipeline based on MACSE v2, an extended version of our codon-aware multiple sequence alignment software accounting for frameshifts and stop codons. The MACSE_BARCODE pipeline allows the accurate alignment of hundreds of thousands of protein-coding barcode sequences. Re-analyses of published data sets confirm that MACSE v2 is able to automatically detect most sequencing errors previously identified manually. The proposed alignment strategy hence alleviates the risk of incorrect species delimitation due the incorporation of sequencing errors or undetected pseudogenes. By applying the MACSE_BARCODE pipeline to mammal, ant, and flowering plant barcode sequences available in BOLD, we highlight several cases of database errors and provide curated reference alignments for the main protein-coding barcode genes. We anticipate our approach to be particularly useful for metabarcoding studies in which thousands of new sequences need to be compared to a reference database for subsequent taxonomic assignment. This might prove particularly helpful for diet characterization studies and large-scale biodiversity assessments through environmental DNA metabarcoding. The new MACSE_BARCODE pipeline is distributed as Nextflow workflows that are available from the MACSE project webpage (`https://bioweb.supagro.inra.fr/macse/`).

How to cite: Frédéric Delsuc and Vincent Ranwez (2020). Accurate alignment of (meta)barcoding data sets using MACSE. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter No. 2.3, pp. 2.3:1–2.3:31. No commercial publisher | Authors open access book. The book is freely available at https://hal.inria.fr/PGE.

[1] FD was funded by the European Research Council via the ERC-2015-CoG-683257 ConvergeAnt project.
[2] VR was supported by the CIRAD UMR AGAP HPC Data Center of the South Green Bioinformatics platform (`http://www.southgreen.fr/`).

Phylogenetics in the genomic era.
Editors: Celine Scornavacca, Frédéric Delsuc and Nicolas Galtier; chapter No. 2.3; pp. 2.3:1–2.3:31
A book completely handled by researchers.
No publisher has been paid.

1 DNA (meta)barcoding and MACSE

Since the birth of molecular systematics in the mid 1960s (Zuckerkandl and Pauling, 1965; Sarich and Wilson, 1967) evolutionary biologists have used molecules to characterize species biodiversity and evolution. The concept of using DNA sequences to distinguish species and reconstruct their phylogenetic relationships based on a universal molecular marker has been adopted early on by microbiologists following the pioneering work of Carl Woese and collaborators on the *16S ribosomal RNA* (*16S rRNA*) gene (Woese and Fox, 1977; Woese et al., 1990). The proposal of using a few standardized universal molecular markers for species identification via the so-called "DNA barcoding" approach has been later popularized by Hebert et al. (2003a). This approach has since been largely embraced by the molecular ecology community and has found many applications from large-scale species inventories (Hebert et al., 2004; Smith et al., 2005; Ward et al., 2005; Lahaye et al., 2008), to more global biodiversity assessments through metabarcoding thanks to the developments of high-throughput sequencing of environmental DNA (Taberlet et al., 2012; Ji et al., 2013; Bohmann et al., 2014).

In practice, almost two decades of DNA barcoding has resulted in the build-up of reference sequence databases of universal barcoding markers linked to biological specimens for fungi, animals, and plants. The Barcode of Life Datasystem (BOLD, Ratnasingham and Hebert 2007) version 4 now contains more than 8 million barcode sequences representing more than 300,000 species. The vast majority of these sequences are from the mitochondrial *cytochrome c oxidase I* (*COI*, Hebert et al. 2003a) gene for animals and from the chloroplastic *Ribulose-1,5-bisphosphate carboxylase/oxygenase* (*rbcL*, Kress and Erickson 2007) and *Maturase K* (*matK*, Dunning and Savolainen 2010) genes for plants. These protein-coding genes have been selected for their ability to discriminate species by showing a clear separation between intraspecific polymorphism and interspecific divergence (Hebert et al., 2003b; CBOL Plant Working Group, 2009). These reference databases offer a great resource and are routinely used to assess the taxonomic assignment of newly produced barcoding sequences in an ever-growing number of barcoding and metabarcoding projects. In light of their importance to the field, quality controls have been introduced for sequence deposit with recommendations on how to validate sequences to be included in the database. Despite these precautions, the BOLD database is not immune to sequencing errors leading to bad sequence quality and undetected pseudogenes (Stoeckle and Kerr, 2012).

One particular problem is the potential integration of nuclear copies of mitochondrial derived genes (*numts*, Lopez et al. 1994) in *COI*, which are known to be widespread in a number of animal groups (Bensasson et al., 2001). Practical solutions have been proposed for limiting the co-amplification of *numts* with mitochondrial barcoding markers (Moulton et al., 2010; Calvignac et al., 2011) but *numts* are difficult to detect in practice and they create obvious problems for species delimitation (Song et al., 2008). For protein-coding barcode genes, alignment to reference sequences, and detection of frameshifts and stop codons are part of the requirements for sequence deposition in BOLD. However, it has been shown that errors in barcoding sequences tend to be clustered at sequence extremities; this illustrates the potential problem of relying only on stop codon detection based on a short fragment within which frameshifts close to extremities will go undetected (Stoeckle and Kerr, 2012). Buhay (2009) highlighted the problems of quality control by examining sequences visually, but the huge number of new DNA barcoding sequences being produced –coupled with the development of metabarcoding– make visual detection and manual correction impossible. Voices have even been raised to question the ability of the DNA barcoding community to embrace high-throughput sequencing (Taylor and Harris, 2012) while others have underlined the bioinformatic challenges associated with the rise of DNA metabarcoding (Coissac et al., 2012).

In this context, our multiple sequence alignment program MACSE (Ranwez et al., 2011), which accounts from frameshifts and stop codons, has been adopted early on by the DNA barcoding community. MACSE is indeed particularly suited to deal with protein-coding barcode sequences that are generally highly conserved at the protein level. Consequently, the first version of MACSE v1 was quickly introduced into metabarcoding bioinformatic pipelines as a denoising step, allowing the automatic detection of stop codons and frameshifts in sequences produced by error-prone sequencing technologies –such as 454 Life Sciences pyrosequencing (Yu et al., 2012; Ji et al., 2013; Ramirez-Gonzalez et al., 2013; Yang et al., 2014). This problem was later alleviated by the development of metabarcoding protocols based on Illumina short read sequencing (Liu et al., 2013). However, third generation long-read sequencing technologies, e.g. Pacific Biosciences and Oxford Nanopore, still suffer from relatively high error rates linked to homopolymers. Besides sequencing error detection, MACSE is also relevant as a tool to automatically spot *numts* and pseudogenes, all the more so as protein-coding markers present numerous advantages as barcoding markers compared to non-coding ones (Andújar et al., 2018).

MACSE is, however, much more than a protein-coding sequence denoising tool. It is first and foremost a multiple sequence alignment program that has recently been enriched with a toolkit of subprograms to handle protein-coding alignments (MACSE v2, Ranwez et al. 2018). A DNA barcoding application for which MACSE has been and could be particularly useful is diet assessment via metabarcoding (Pompanon et al., 2012). This kind of study typically requires thousands of newly produced barcoding sequences to be compared against a reference database such as BOLD to perform taxonomic assignment of prey items. In this case, as in most other metabarcoding applications (Leray and Knowlton, 2015), accurately aligning the newly produced sequences to the sequences of the reference database could be particularly valuable. MACSE has effectively been used for doing so in the context of the Moorea Biocode project, which aimed to create the first comprehensive inventory of all non-microbial life in a complex tropical ecosystem (Leray et al., 2013). Indeed, working from a multiple sequence alignment allows to leverage the power of phylogenetics for taxonomic assignment instead of relying on simple sequence similarity searches. The advantages of using alignments and phylogenetic trees have long been recognized in the microbiome field, in which the main dedicated pipelines based on *16S rRNA* metabarcoding, such as MOTHUR (Schloss et al., 2009) and QIIME (Caporaso et al., 2010), are routinely used to performed taxonomic assignment based on curated databases of sequence alignments and phylogenetic trees with updated taxonomy such as Greengenes (DeSantis et al., 2006) and SILVA (Pruesse et al., 2007). Ultimately, with the availability of reference alignments and phylogenetic trees, taxonomic assignment could be based on probabilistic evolutionary placement as implemented in programs such as pplacer (Matsen et al., 2010), RAxML_EPA (Berger et al., 2011), and RAPPAS (Linard et al., 2019). Unfortunately, it is far from being the case in the DNA barcoding field, as no such reference alignments and trees exist. Indeed, if public sequence data can easily be downloaded from the BOLD database for the different taxonomic groups represented, the resulting files contain unaligned sequences consisting of a mix of different barcoding fragments and markers.

In this chapter, after illustrating the usefulness of MACSE v2 to deal with sequencing errors and *numts* in barcoding data sets, we present a new pipeline named MACSE_BARCODE, which aims at accurately aligning hundreds of thousands of protein-coding barcode sequences.

By applying MACSE_BARCODE to mammal, ant, and flowering plant sequences of the BOLD database, we provide high quality reference alignments of *COI* (mammals and ants), and *rbcL* and *matK* (flowering plants) as freely available resources for the DNA barcoding community.

2 Using MACSE to align protein-coding (meta)barcoding data sets

In this section, we provide a short overview of the features included in MACSE v2, with an emphasis on the subprograms and options that are particularly useful when dealing with protein-coding barcoding data. The MACSE_BARCODE pipeline is based on these subprograms and could be run in two command lines without *a priori* knowledge of the underlying details. However, getting familiar with the subprograms at the core of the barcoding pipeline will allow the user to understand what is in the blackbox and, thus, help with the interpretation of the results. A more detailed presentation of MACSE v2 can be found in Ranwez et al. (2020).

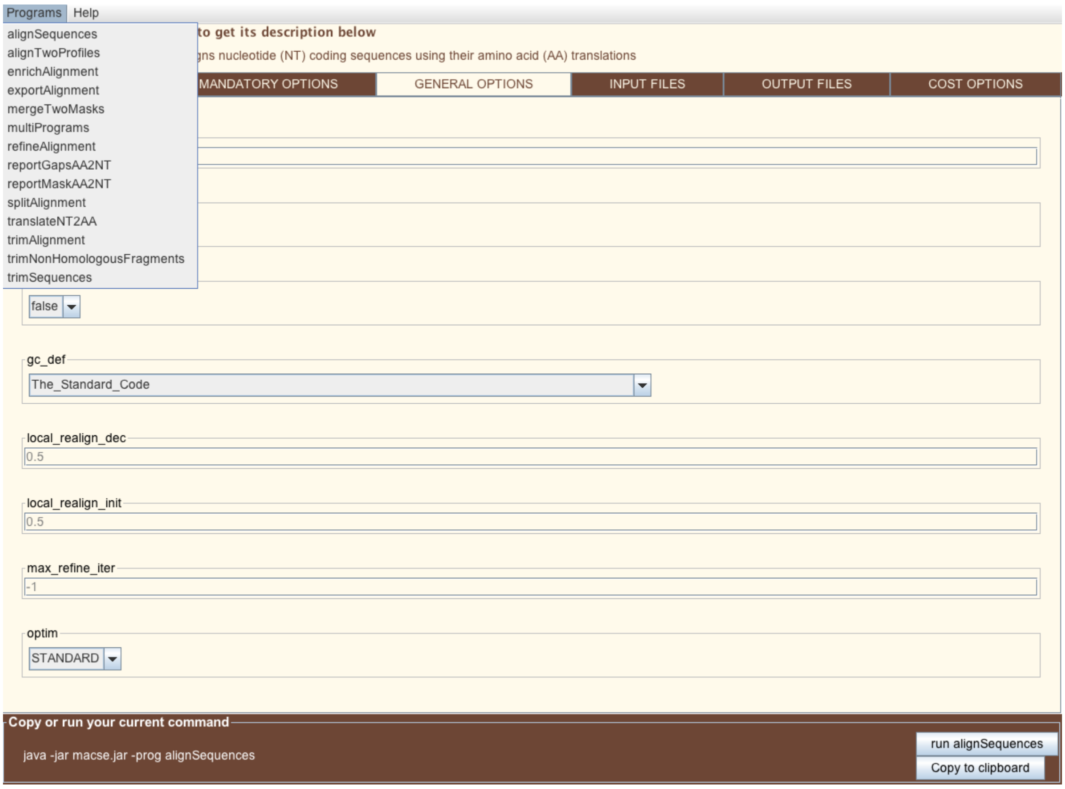
2.1 A quick overview of MACSE v2

MACSE v1 (Ranwez et al., 2011) was originally developed as a single program to align nucleotide sequences of protein-coding genes while accounting for frameshifts and stop codons. Its second version (MACSE v2, Ranwez et al. 2018) extended the initial release with a suite of subprograms and a Graphical User Interface (GUI). The main subprogram is *alignSequences*, which is at the core of MACSE. The other subprograms constitute a rich toolkit for handling alignments of protein-coding sequences. The current MACSE version v2.03 includes 14 subprograms that, for instance, allow: (1) refining an existing alignment (*refineAlignment*), (2) trimming an existing alignment (*trimAlignment*), (3) removing non-homologous sequence fragments (*trimNonhomologousFragments*), (4) excluding nucleotides, codons, or sites involved in frameshifts and stop codons (*exportAlignment*), (5) merging two distinct alignments (*alignTwoProfiles*), and (6) enriching an alignment by sequentially adding sequences (*enrichAlignment*). These subprograms can thus be combined or sequentially executed to design simple alignment pipelines. All analyses in this chapter were performed using the command line version of the latest MACSE release (but note that the analyses that do not involve the metabarcoding pipeline could also be reproduced using the GUI, see Figure 1).

MACSE can be freely downloaded as a single jar file from its dedicated website (`https://bioweb.supagro.inra.fr/macse/`). As it is written in the JAVA language, MACSE does not need installation and should run under any operating system (Windows, MAC OS, Linux), provided that the Java Runtime Environment (JRE) is installed on the computer. The MACSE website contains detailed documentation and tutorials presenting several examples of applications. Once downloaded, MACSE can be directly launched by double clicking on the `macse_v2.03.jar` file or by typing the following command in a terminal:

```
$ java -jar ./macse_v2.03.jar
```

In both cases, this should open the GUI version of MACSE (Figure 1). The “Programs” menu allows choosing from the different subprograms that were designed to align and manipulate protein-coding sequence `.fasta` files. Once a subprogram is selected, a brief description of this subprogram is provided and the different options are available from the different tabs corresponding to mandatory options, e.g. output file names, alignment parameters, etc... When an option field is selected by clicking on it, the related documentation



to get its description below

gns nucleotide (NT) coding sequences using their amino acid (AA) translations

false

gc_def
The_Standard_Code

local_realign_dec
0.5

local_realign_init
0.5

max_refine_iter
-1

optim
STANDARD

Copy or run your current command

```
java -jar macse.jar -prog alignSequences
```

Figure 1 An overview of MACSE v2 Graphical User Interface (GUI) showing the list of available subprograms in the "Programs" menu.

is displayed. Moreover, the corresponding command line appears at the bottom of the GUI and can be directly copied to the clipboard to run the same analysis via the command line and ensure the traceability of the analysis.

The following command launches the command line version of MACSE and prints a help message listing all valid subprograms with a one-line description for each of them:

```
$ java -jar ./macse_v2.03.jar -help
```

The `-prog` option allows users to specify the subprogram to be executed; without any further specification, the following command will print a basic help message for the *alignSequences* subprogram describing its mandatory options:

```
$ java -jar ./macse_v2.03.jar -prog alignSequences
```

Adding the `-help` option will print a help message with more detailed information on the subprogram and the complete list of its available options:

```
$ java -jar ./macse_v2.03.jar -prog alignSequences -help
```

As MACSE is run through the Java virtual machine, the memory that Java is allowed to use can be increased via the `-Xmx` option. This is not a MACSE option *per se*, but it is definitely essential for dealing with relatively large data sets. The following command will for instance allocate 600 MB to the Java virtual machine to run the *alignSequences* subprogram:

```
$ java -jar ./macse_v2.03.jar -Xmx 600m -prog alignSequences
```

2.3:6 Metabarcoding alignements using MACSE

The most basic usage of MACSE to align protein-coding sequences is to use the *alignSequences* subprogram with a `.fasta` file containing protein-coding nucleotide sequences. As *alignSequences* relies on amino acid translations to align sequences, the input nucleotide sequences need to be all in the same forward direction (no reverse complement sequences) and should consist of an open reading frame (ORF) for most of their length (no UTR or intron fragments). The following command will align the 20 mammalian sequences of the *TMEM184a* nuclear gene contained in the `tmem184a.fasta` file (available from the MACSE online tutorial) with default parameters:

```
$ java -jar ./macse_v2.03.jar -prog alignSequences -seq tmem184a.fasta
```

This command will generate two `.fasta` files respectively containing the protein-coding nucleotide sequences aligned as codons and the corresponding amino acid alignment (Figure 2). In this example, MACSE detected frameshifts indicated by exclamation marks ('!') in both the dolphin (*Tursiops*) and the orang-utan (*Pongo*) *TMEM184a* nucleotide sequences. These frameshifts stem from the same two nucleotide deletions and likely correspond to sequencing or annotation errors of this gene in these two species. As MACSE allows aligning these protein-coding sequences by conserving the coding frame, the incomplete codons containing the inferred frameshifts are directly translated into exclamation marks ('!') in the corresponding amino acid alignment.

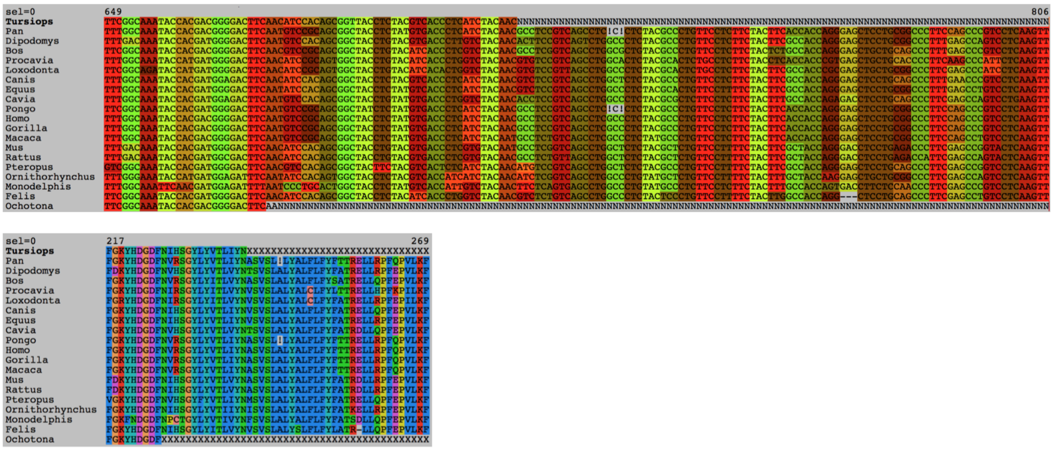


Figure 2 Excerpts of the output nucleotide ("_NT") and amino acid ("_AA") alignments of 20 mammalian sequences of the *TMEM184a* nuclear gene obtained with MACSE subprogram *alignSequences* basic usage and visualized using SeaView (Gouy et al., 2010). Note the frameshifts indicated by exclamation marks ('!') inferred by MACSE in the dolphin (*Tursiops*) and orang-utan (*Pongo*) codon sequences and in the corresponding amino acid alignment.

By default, the names of the output files are based on the input file name by adding the "_NT" and "_AA" suffixes for nucleotides and amino acids, respectively. Custom output file names can be specified using the `-out_NT` and `-out_AA` options. MACSE relies on amino acid translation and uses the standard genetic code by default. Other genetic codes could be specified using the `-gc_def` option following the NCBI nomenclature (`https://www.ncbi.nlm.nih.gov/Taxonomy/Utils/wprintgc.cgi`). For instance, the invertebrate mitochondrial code is number 5. The following command allows aligning mitochondrial *COI* sequences of grasshoppers:

```
$ java -jar ./macse_v2.03.jar
```

```
        -prog alignSequences
        -seq Grasshoppers_COI.fasta
        -gc_def 5
        -out_NT Grasshoppers_COI_NT.fasta
        -out_AA Grasshoppers_COI_AA.fasta
```

If the data set contains sequences with different genetic codes, as it could be the case, for instance, in mitochondrial *COI* barcoding data from multiple animal phyla, they could be specified in a separated text file using the `-gc_file` option. This file should indicate, on each line, the sequence names with their corresponding genetic code numbers. The following command will allow aligning metazoan *COI* sequences extracted from Singh et al. (2009) including five different genetic codes:

```
$ java -jar ./macse_v2.03.jar
          -prog alignSequences
          -seq Singh2009_COI.fasta
          -gc_file Singh2009_COI_gc_file.txt
          -out_NT Singh2009_COI_NT.fasta
          -out_AA Singh2009_COI_AA.fasta
```

A key feature of MACSE for aligning protein-coding barcoding data sets, which are by essence well conserved at the amino acid level, concerns the way "cost parameters" can be fine tuned, in particular the costs associated with frameshifts (`-fs` option) and stop codons (`-stop` option). As in most multiple sequence alignment software, the ratio between gap extension cost (`-gap_ext` option) and gap opening cost (`-gap_op` option) can be specified in MACSE. However, MACSE also allows adjustment of the relative cost of gaps appearing at the sequence extremities (`-gap_op_term` and `-gap_ext_term` options). By default, external gaps are less penalized as they often reflect the fact that a sequence was partially sequenced. Similarly, the occurrence of one or two missing nucleotides at the sequence extremities lead to incomplete codons; but such external frameshifts (`-fs_term` option) should not be as penalized as those occurring in the middle of a sequence. Moreover, when a data set contains a mix of functional and pseudogene sequences, or sequences of variable sequencing qualities (e.g. reference genome sequences and *de novo* assembled contigs), it might be relevant to assign different penalties for the frameshifts (`-fs_lr` option) and stop codons (`-stop_lr` option) appearing in the less reliable sequences. In such cases, MACSE allows the user to *a priori* define two sets of sequences by providing two `.fasta` files as input instead of a single one. The most reliable sequences are in the file provided by the `-seq` option, whereas the least reliable ones are in the file provided by the `-seq_lr` option. The default cost parameters usually work fine, but guidelines to help adjusting parameter costs for some specific types of sequence data sets are provided in the MACSE online documentation. The default values for each parameter can be explored through the GUI.

In the context of metabarcoding studies, one of the main challenges is to add thousands of newly generated barcoding sequences to a reference database for subsequent taxonomic assignment. MACSE v1 was successfully used to implement such an approach in the context of the Moorea Biocode project for characterizing coral reef fish gut contents based on *COI* metabarcoding (Leray et al., 2013). The newly developed *enrichAlignment* subprogram of MACSE v2 can now be used to sequentially enrich a reference alignment with newly generated sequences, possibly adapting the cost parameters for the latter. This subprogram also contains specific options to control the quality of the sequences to be added. For instance, the following command will sequentially enrich a reference alignment of *COI* arthropod sequences by adding only newly generated *COI* fragments obtained from fish gut content that do not induce too many frameshifts (-maxFS_inSeq option), stop codons (-maxSTOP_inSeq option), and insertions (--maxINS_inSeq option):

```
$ java -jar ./macse_v2.03.jar
          -prog enrichAlignment
          -align Moorea_BIOCODE_small_ref.fasta
          -seq Moorea_BIOCODE_small_ref.fasta
          -seq_lr noctural_diet_sample.fasta
          -gc_def 5
          -fs_lr 10
          -stop_lr 10
          -maxFS_inSeq 0
          -maxINS_inSeq 0
          -maxSTOP_inSeq 1
```

In addition to the two usual output files –respectively containing the final enriched nucleotide ("_NT") and amino acid alignments ("_AA")– the *enrichAlignment* subprogram provides a tabular text file providing detailed statistics for each target sequence, including how many stop codons, frameshifts, and insertion events were required to align it with the reference alignment, and whether it has been added or not based on the specified criteria.

2.2 MACSE_BARCODE: An efficient metabarcoding alignment pipeline

Metabarcoding analysis often requires dealing with several thousands of sequences. Such data sets are not directly tractable with the *alignSequence* subprogram of MACSE, nor by any other classical multiple sequence alignment program (see Chapter 2.2 [Ranwez and Chantret 2020]). However, they can be handled by sequentially adding each barcoding sequence to a reference alignment containing sequences of the targeted locus, as we successfully implemented in the Moorea Biocode project (Leray et al., 2013). Adding sequence one by one is not a second-best option; this strategy has been suggested to produce high quality alignments when dealing with thousands of sequences (Boyce et al., 2014). However, even this strategy could be time consuming when dealing with hundreds of thousands of sequences. Fortunately, at a low taxonomic scale, barcoding sequences are quite similar and contain few indels, if any. As a consequence, if the reference alignment captures most of the sequence diversity, most sequences can be aligned against it without inducing new gap sites in the alignment. This particularity allows the parallelization of the alignment process in which each sequence can be separately aligned to the reference alignment. All sequences that can be aligned to the reference alignment without insertion events can then be combined to build a large alignment containing most sequences of the input data set. This can easily be done using the *enrichAlignment* subprogram of MACSE with the --maxINS_inSeq 0 option. The remaining sequences can eventually be added afterwards using the same subprogram by relaxing the assumption of no insertion.

The proposed approach implemented in the MACSE_BARCODE pipeline consists of three steps. First, identifying a small subset of a few hundred sequences that best represent the barcoding data set diversity. Second, aligning these representative sequences to build a reference alignment. Third, using this reference alignment to align the thousands of remaining barcode sequences. Online documentation related to MACSE-based pipelines can be found at: https://bioweb.supagro.inra.fr/macse/index.php?menu=docPipeline/docPipelineHtml.

2.2.1 Identifying a small subset of representative sequences

Barcoding data sets may contain non-homologous sequences, as well as reverse complement sequences. To correctly handle these cases, we rely on a carefully chosen reference nucleotide sequence that should translate perfectly into amino acids from start to end. Our pipeline to identify representative sequences (Figure 3) takes advantage of MMseqs2 (Steinegger and Söding, 2017) and takes three key inputs: (1) the fasta file of all barcoding sequences to be aligned, (2) the reference nucleotide sequence, and (3) the genetic code used for translating these sequences. Given these three inputs, the pipeline proceeds as follows:

- Step 1: **MMseqs2 easy-search** is used to compare the translation of each barcode sequence in the six possible reading frames with the translation of the reference nucleotide sequence obtained with the *translateNT2AA* subprogram of MACSE v2. This produces a set of amino acid sequences similar to the reference, and a result table that summarizes all these comparisons in terms of amino acid similarity.
- Step 2: **MMseqs2 easy-cluster** is used to cluster this set of amino acid sequences. The clusters are then sorted based on the number of sequences they contain. The centroid sequences of the largest clusters are identified and their nucleotide sequences are extracted to form the set of representative sequences. The clustering is performed using a strict criterion of 100% amino acid sequence identity. This script has two tuning parameters that control the number of representative sequences. The first one `--in_minClustSize` allows specifying the minimal number of sequences a cluster must contain to be considered (default value = 10). The second one `--in_maxRepresentativeSeqs` allows specifying the maximum number of relevant sequences in the output (default value = 100).
- Step 3: the output of the **MMseqs2 easy-search** step is processed to identify all input sequences that are homologous to the reference sequence and to reverse complement them using seqtk (`https://github.com/lh3/seqtk`), if needed.

The **representative_seqs** sub-pipeline is provided as a Singularity container (Kurtzer et al., 2017) and can be built from the receipt file available from the MACSE_V2_PIPELINES page (`https://github.com/ranwez/MACSE_V2_PIPELINES/tree/master/MACSE_BARCODE`), or directly downloaded from the Singularity official library (Sochat et al., 2017) using the following command:

```
$ singularity pull
        --arch amd64 library://vranwez/default/representative_seqs:v01
```

As for all our Singularity based pipelines (Ranwez et al., 2020), the help message can be obtained by launching them without parameters using the following command:

```
$ ./representative_seqs_v01.sif
```

To launch the identification of the representative sequences for the Magnoliophyta *matK* data set (see Section 3 below) the command line is:

```
$ ./representative_seqs_v01.sif
        --in_refSeq  Magnolia_officinalis_NC_020316.1_matK_ref.fasta
        --in_genetic_code 11
        --in_seqFile Magnoliophyta_BOLD_matK_107413seqs.fasta
        --out_repSeq Magnoliophyta_matK_repSeq.fasta
        --out_homologSeq Magnoliophyta_matK_homologous.fasta
        --out_listRevComp Magnoliophyta_matK_revComp.list
        --in_minClustSize 10
        --in_maxRepresentativeSeqs 100
```

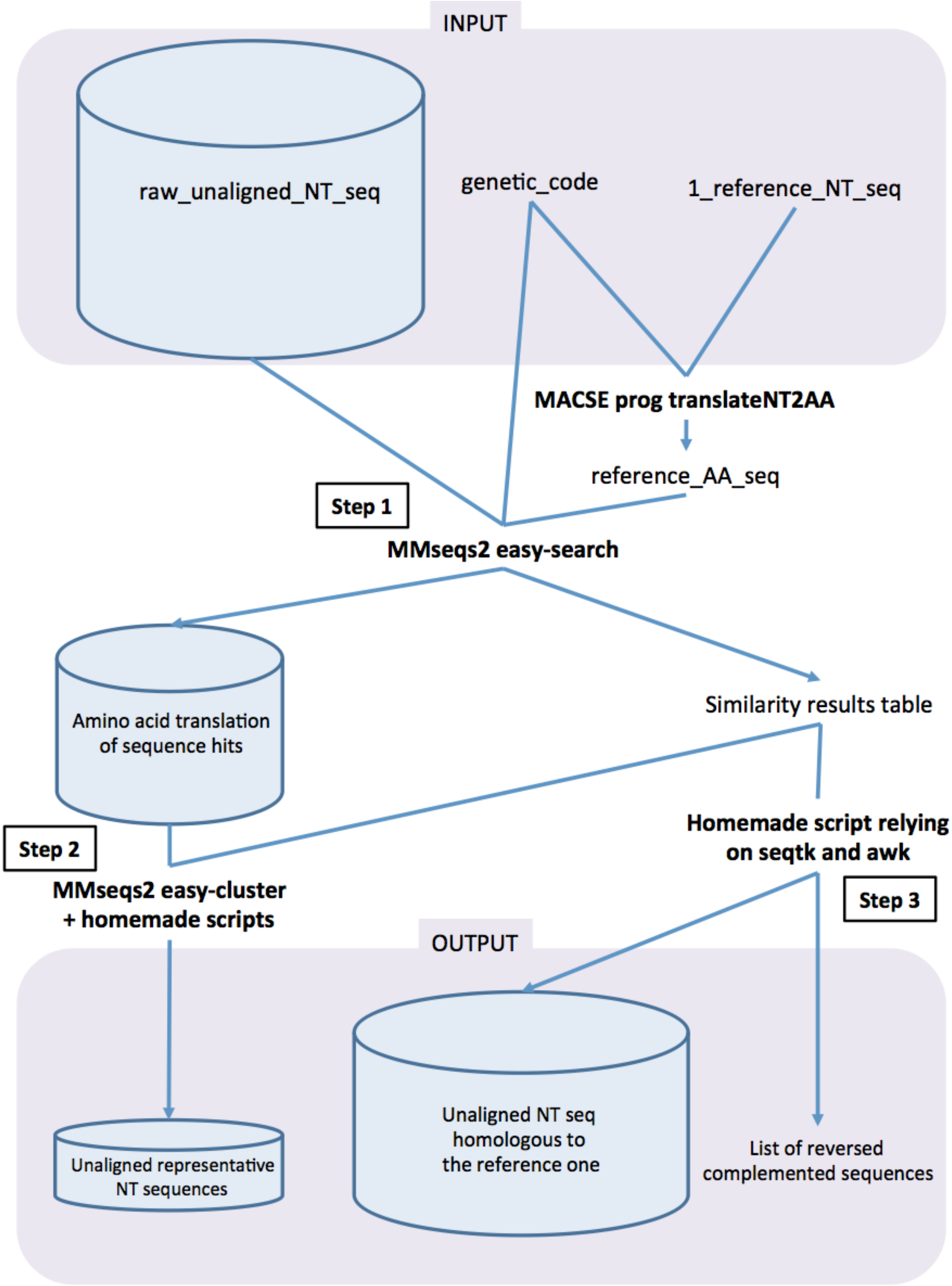

INPUT
raw_unaligned_NT_seq
genetic_code
1_reference_NT_seq
MACSE prog translateNT2AA
reference_AA_seq
Step 1
MMseqs2 easy-search
Amino acid translation of sequence hits
Similarity results table
Step 2
MMseqs2 easy-cluster + homemade scripts
Homemade script relying on seqtk and awk
Step 3
OUTPUT
Unaligned representative NT sequences
Unaligned NT seq homologous to the reference one
List of reversed complemented sequences

Figure 3 Overview of the representative-sequences identification step of the MACSE_BARCODE pipeline as implemented in the **representative_seqs** Singularity container.

2.2.2 Aligning relevant sequences to get a representative alignment

The relevant nucleotide sequences identified at the previous step could be aligned using the OMM_MACSE pipeline. This pipeline was initially designed to be able to rapidly infer the thousands of CDS alignments of the 10th release of the OrthoMaM database (Scornavacca et al., 2019). This pipeline relies on the amino acid translations to align the coding sequences and includes several optional filtering steps and is more extensively described in Ranwez et al. (2020). It is also provided as a Singularity container and can be built from the receipt file available on our github page or directly downloaded from the Singularity official library using the following command:

```
$ singularity pull
        --arch amd64 library://vranwez/default/omm_macse:v10.02
```

For this specific task, we advise to use the reference nucleotide sequence as the unique "reliable sequence" and to provide the subset of relevant nucleotide sequences as "less reliable sequences". This should help to identify the correct reading frame in case some of the relevant sequences do not start on the first reading frame. We also suggest to avoid the pre-filtering, and alignment trimming steps to ensure that the reference sequence is preserved completely even if the relevant sequences correspond only to a specific sub-fragment:

```
$ ./omm_macse_v10.02.sif
        --in_seq_file Magnolia_officinalis_NC_020316.1_matK_ref.fasta
        --in_seq_lr_file Magnoliophyta_matK_repSeq.fasta
        --out_dir REF_ALIGN_Magnoliophyta_matK
        --out_file_prefix Magnoliophyta_matK
        --genetic_code_number 11
        --no_prefiltering
        --min_percent_NT_at_ends 0
        --java_mem  200m
```

To simplify the process, we provide a Nextflow workflow called **P_buildRefAlignment** that chains these two first steps to directly build the alignment of representative sequences. Nextflow (Di Tommaso et al., 2017) enables scalable and reproducible scientific workflows using software containers allowing the adaptation of pipelines written in the most common scripting languages. It could easily be installed using the following linux commands:

```
$ curl -s https://get.nextflow.io | bash
```

or

```
$ wget -qO- https://get.nextflow.io | bash
```

To launch the **P_buildRefAlignment** workflow, the two previously described Singularity containers **representative_seqs** and **omm_macse** are needed and the "nextflow.config" file should be adapted to the computer environment.

Nextflow separates the workflow itself from the directive regarding the correct way to execute it in the environment. By default, if no "nextflow.config" file is provided, the workflow is launched as a single process on the current machine. One key advantage of Nextflow is that, by changing slightly the "nextflow.config" file, the same workflow will be parallelized and launched to exploit the full resources of a high performance computing (HPC) cluster. The key parameters to change in this configuration file are: (1) the "executor", which could be "local" to run on a standard machine, "sge" or "slurm" to be launched on a HPC cluster or even run on the cloud, and (2) the "queue", which specifies on which queue the job should be run if a grid based executor is used. An example “nextflow.config” file is provided on the MACSE_BARCODE github page. For instance, the alignment of representative sequences for the Magnoliophyta *matK* data set could be directly built by just running the following command:

```
$ ./nextflow P_buildRefAlignment.nf
        --refSeq Magnolia_officinalis_NC_020316.1_matK_ref.fasta
        --seqToAlign Magnoliophyta_BOLD_matK_107413seqs.fasta
        --geneticCode 11
        --outPrefix Magnoliophyta_matK
```

Note that, despite the care taken in the construction of this pipeline, the output reference alignment may still contain errors. We thus strongly advise to carefully check the resulting alignment, and to remove spurious sequences, if present, before using it as a reference alignment for aligning the remaining thousands of barcode sequences.

2.2.3 Aligning thousands of barcoding sequences using a reference alignment

In order to align the remaining barcode sequences to the reference alignment (Figure 4), a second Nextflow workflow named **P__enrichAlignment** automatizes the two following steps for each sequence. First, each targeted barcoding sequence is compared with the reference alignment and its extremities are trimmed if they are not homologous to the reference alignment. This step allows removing 5’ or 3’ sequence extremities that do not correspond to the target barcoding locus and, if kept, would impede the sequence to be added without insertion events. Second, the trimmed version of each barcoding sequence is aligned with the reference alignment. The number of unexpected frameshifts, stop codons, and insertion events observed in the aligned sequence is counted and saved in a report `.csv` file, which allows to know exactly the reasons why each sequence has been kept, or not, in the final alignment. By default, the sequences present in the final alignment are those that can be aligned with the reference alignment while having at most two internal frameshifts (incomplete codons at 5’ and 3’ ends of a sequence are not penalized) and one internal stop codon (a stop codon as the final codon is not penalized). This corresponds to the following options of the *enrichAlignment* subprogram of MACSE `-maxFS_inSeq 2 -maxINS_inSeq 0 -maxSTOP_inSeq 1`. It is easy to spot these parameters in the Nextflow pipeline and to change them if needed. The only parameter that should not be changed is `-maxINS_inSeq 0` as it is a prerequisite for the parallelization. Basically, this workflow (summarized in Figure 4) simply splits the large input data set in small subsets of 100 sequences that are treated in parallel (trimmed then aligned) before concatenating the obtained result files.

To execute this workflow, the previously computed reference alignment, the set of homologous barcode sequences to be aligned, and the genetic code to be used should be provided, as well as a prefix for the output file names using the following command:

```
$ ./nextflow P_enrichAlignment.nf
        --refAlign Magnoliophyta_MATK_reference_alignment_NT.aln
        --seqToAlign Magnoliophyta_MATK_homolog.fasta
        --geneticCode 11
        --outPrefix Magnoliophyta_MATK
```

Finally, if running the **P__buildRefAlignment** and **P__enrichAlignment** workflows separately is advisable to check the identification of the representative sequences and the construction of the reference alignment, we also provide a third Nextflow workflow named

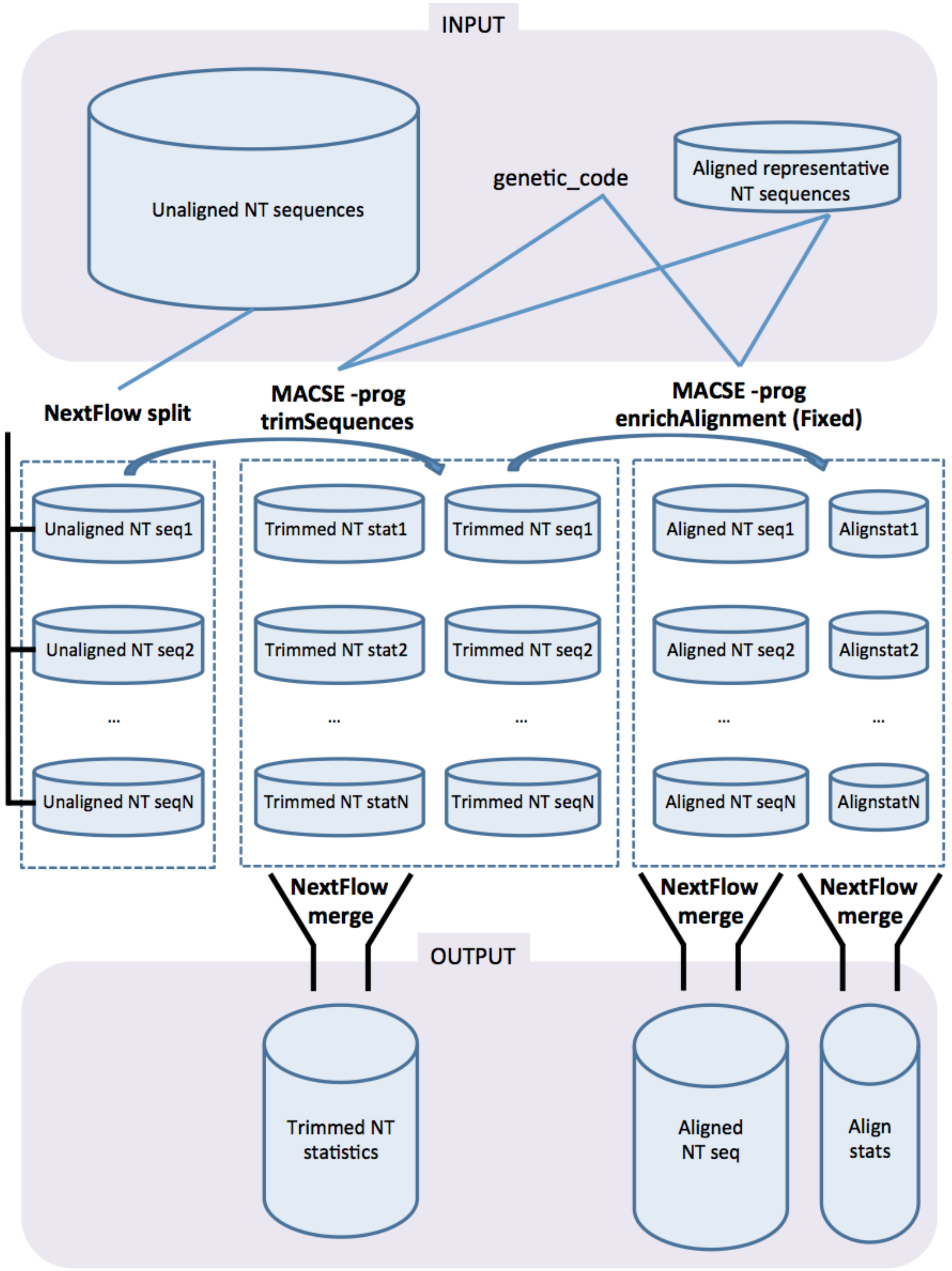


Figure 4 Overview of the parallel enrich alignment step of the MACSE_BARCODE pipeline as implemented in the **P__enrichAlignment** Nextflow workflow.

P__macse__barcode to execute the whole MACSE_BARCODE pipeline. This workflow could be simply executed on the Magnoliophyta *matK* data set by providing the reference sequence, the initial set of barcode sequences to be aligned, the genetic code, and a prefix for the output file names using the following command:

```
$ ./nextflow P_macse_barcode.nf
        --refSeq Magnolia_officinalis_NC_020316.1_matK_ref.fasta
        --seqToAlign Magnoliophyta_BOLD_matK_107413seqs.fasta
        --geneticCode 11
        --outPrefix Magnoliophyta_matK
```

3 (Meta)barcoding case studies

3.1 MACSE automatically detects sequencing errors in *COI* sequences

3.1.1 Bad quality crayfish *COI* sequences

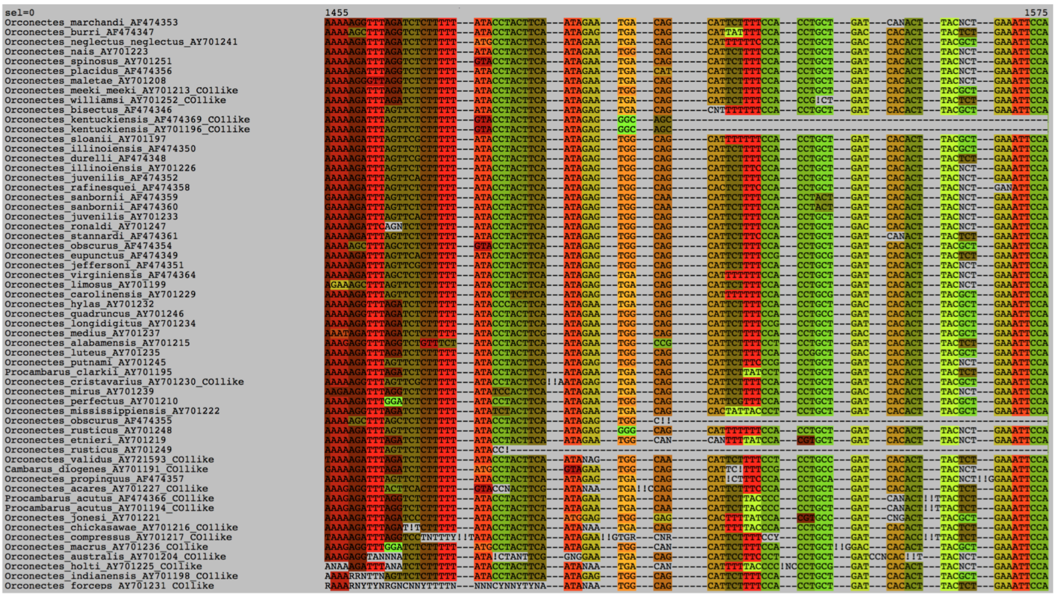
The first example data set (TK2006) is an alignment of 89 crayfish *COI* sequences produced by Taylor and Knouft (2006). This data set includes 24 poor quality sequences that have been flagged as "COI-like" by GenBank after submission as it has been pointed out by Buhay (2009), who later provided a manually curated alignment for these sequences. We used this data set to evaluate the capability of MACSE to automatically provide an accurate alignment of a set of sequences containing both correct and incorrect *COI* sequences. As was done manually by Buhay (2009), we used the *COI* sequence extracted from the complete mitochondrial genome of *Cherax destructor* (NC_011243) as a reference to automatically align the TK2006 data set using the *alignSequences* subprogram of MACSE. We used differential costs for frameshifts (`-fs 30`) and stop codons (`-stop 30`) for the reference sequence, and `-fs_lr 10` and `-stop_lr 10` for all other sequences considered as less reliable including "COI-like" sequences:

```
$ java -jar ./macse_v2.03.jar
        -prog alignSequences
        -gc_def 5
        -seq Cherax_destructor_ref.fasta
        -seq_lr TK2006.fasta
        -fs 30
        -stop 30
        -fs_lr 10
        -stop_lr 10
        -out_NT TK2006_macse_NT.fasta
        -out_AA TK2006_macse_AA.fasta
```

The resulting nucleotide and amino acid alignments can be visually inspected using SeaView, which handles the exclamation mark ('!') character used by MACSE to identify frameshifts and also allows coloring the alignment by codons (Figure 5).

The statistics on the number of frameshifts and stop codons inferred in each sequence of the resulting nucleotide alignment can be output in a `.csv` file (`--out_stat_per_seq` option) using the *exportAlignment* subprogram with the following command:

```
$ java -jar ./macse_v2.03.jar
        -prog exportAlignment
        -gc_def 5
        -align TK2006_macse_NT.fasta
        -out_stat_per_seq TK2006_macse_NT_stat.csv
```



```
sel=0                                     1455                                                                                                                        1575
Orconectes_marchandi_AF474353             AAAAAGGTTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAT---CANACT---TACNCT---GAAATTCCA
Orconectes_burri_AF474347                 AAAAAGCTTTAGGTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGA---CAG------CATTATTTTCCA---CCTGCT---GAC---CACACT---TACTCT---GAAATTCCA
Orconectes_neglectus_neglectus_AY701241   AAAAAGATTTAGATCTCTTTTT---ATGCCTACTTCA---ATAGAA---TGA---CAG------CATTTTTTCCCA---CCTGCT---GAC---CACACT---TACGCT---GAAATTCCA
Orconectes_nais_AY701223                  AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_spinosus_AY701251              AAAAAGATTTAGGTCTCTTTTT---GTACCTACTTCA---ATAGAA---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_placidus_AF474356              AAAAAGATTTAGGTCTCTTTTT---ATACCTACTTCA---ATAGAA---TGA---CAT------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_maletae_AY701208               AAAAAGGGTTAGGTCTCTTTTT---ATACCTACTTCA---ATAGAA---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_meeki_meeki_AY701213_CO1like   AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_williamsi_AY701252_CO1like     AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGA---CAG------CATTCTTTTCCA---CCG!CT---GAT---CACACT---TACTCT---GAAATTCCA
Orconectes_bisectus_AF474346              AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGA---CAG------CNTTTTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_kentuckiensis_AF474369_CO1like AAAAAGATTTAGTTCTCTTTTT---GTACCTACTTCA---ATAGAG---GGC---AGC---------------------------------------------------------------
Orconectes_kentuckiensis_AY701196_CO1like AAAAAGATTTAGTTCTCTTTTT---GTACCTACTTCA---ATAGAG---GGC---AGC---------------------------------------------------------------
Orconectes_sloanii_AY701197               AAAAAGATTTAGTTCGCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTTTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_illinoiensis_AF474350          AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_durelli_AF474348               AAAAAGATTTAGTTCGCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACTCT---GAAATTCCA
Orconectes_illinoiensis_AY701226          AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTCCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_juvenilis_AF474352             AAAAAGATTTAGTTCACTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_rafinesquei_AF474358           AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CACTCTTTCCCA---CCTGCT---GAT---CACACT---TACNCT---GANATTCCA
Orconectes_sanbornii_AF474359             GAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAA------CATTCTTTTCCA---CCTACT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_sanbornii_AF474360             GAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAA------CATTCTTTTCCA---CCTACT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_juvenilis_AY701233             AAAAAGATTTAGTTCACTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_ronaldi_AY701247               AAAAAGATTCAGNTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAC---CACACT---TACNCT---GAAATTCCA
Orconectes_stannardi_AF474361             AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTCCCA---CCTGCT---GAT---CANACT---TACTCT---GAAATTCCA
Orconectes_obscurus_AF474354              AAAAAGCTTTAGCTCTCTTTTT---GTACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_eupunctus_AF474349             AAAAAGATTTAGTTCACTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCG---CCTGCG---GAT---CACACT---TACTCT---GAAATTCCA
Orconectes_jeffersoni_AF474351            AAAAAGATTTAGTTCGCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_virginiensis_AF474364          AAAAAGATTTAGCTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGA---CAG------CATTTTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_limosus_AY701199               AGAAAGCTTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_carolinensis_AY701229          AAAAAGATTTAGTTCTCTTTTT---ATACCTTCTTCA---ATAGAG---TGA---CAG------CATTTTTTTCCA---CCTGCG---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_hylas_AY701232                 AAAAAGGTTTAGATCTCTTTTT---ATACCCACTTCA---ATAGAA---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAC---CACACT---TACGCT---GAAATTCCA
Orconectes_quadruncus_AY701246            AAAAAGGTTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAG------CATTCTTTTCCA---CCTGCC---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_longidigitus_AY701234          AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAG------CATTCTTTTCCG---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_medius_AY701237                AAATAGATTTAGATCTCTTTTT---ATACCTACTTCG---ATAGAA---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAC---CACACT---TACGCT---GAAATTCCA
Orconectes_alabamensis_AY701215           AAAGAGGTTTAGGTCTGTTTCT---ATACCTACTTCA---ATAGAA---TGG---CCG------CATTCTTTTCCT---CCTGCT---GAT---CACACT---TACTCT---GAAATTCCA
Orconectes_luteus_AY701235                AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAG------CATTCTTTTCCA---CCGGCT---GAC---CACACT---TACGCT---GAAATTCCA
Orconectes_putnami_AY701245               AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTCCCG---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Procambarus_clarkii_AY701195              AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCT---ATAGAA---TGA---CAG------CATTCTTATCCT---CCTGCT---GAT---CACACT---TACTCT---GAAATTCCA
Orconectes_cristavarius_AY701230_CO1like  AAAAAGGTTTAGTTCGCTTTTT---ATACCTACTTCG!!AATAGAG---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_mirus_AY701239                 AAGAAGATTTAGGTCTCTTTTT---ATATCCACTTCA---ATAGAA---TGA---CAG------CATTCTTTTCCA---CCTGCT---GAC---CACACT---TACNCT---GAAATTCCA
Orconectes_perfectus_AY701210             AAAAAGATTTGGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAG------CATTCGTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_mississippiensis_AY701222      AAAAAGGTTTAGATCTCTTTTT---ATATCTACTTCA---ATAGAA---TGA---CAG------CACTATTACCCT---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_obscurus_AF474355              AAAAAGCTTTAGCTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---C!!---------------------------------------------------------------
Orconectes_rusticus_AY701248              AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---GGG---CAG------CATTTTTTTCCA---CCTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_etnieri_AY701219               AAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATAGAA---TGG---CAN------CANTTTTATCCA---CGTGCT---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_rusticus_AY701249              AAAAAGATTTAGTTCTCTTTTT---ATACC!------------------------------------------------------------------------------------------
Orconectes_validus_AY721593_CO1like       TAAAAGATTTAGATCTCTTTTT---ATACCTACTTCA---ATANAG---TGG---CAA------CATTCTTTTCCT---CCTGCC---GAT---CACACT---TACTCT---GAAATTCCA
Cambarus_diogenes_AY701191_CO1like        GAAAAGATTTAGATCTCTTTTT---ATGCCTACTTCA---GTAGAA---TGG---CAG------CATTC!TTTCCG---CCTGCC---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_propinquus_AF474357            AAAAAGATTTAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGA---CAG------CAT!CTTTCCCA---CCTGCA---GAT---CACACT---TACNCT!!GGAAATTCCA
Orconectes_acares_AY701227_CO1like        AAAGAGGTTTACTTCACTTTTT---GTACCNACTTCG---ATANAA---TGA!!CCAG------CATTCTTTCCCA---CCTGCT---GAT---CACACT---TACTCT---GAAATTCCA
Procambarus_acutus_AF474366_CO1like       AAAGAGATTTAGGTCTCTTTTT---ATACCTACTTCT---ATAGAA---TGA---CAA------CATTCTTACCCC---CCTGCT---GAT---CANACT!!TTACTCT---GAAATTCCA
Procambarus_acutus_AY701194_CO1like       AAAGAGATTTAGGTCTCTTTTT---ATACCTACTTCT---ATAGAA---TGA---CAA------CATTCTTACCCC---CCTGCT---GAT---CANACT!!TTACTCT---GAAATTCCA
Orconectes_jonesi_AY701221                AAAAAGATTTAGATCCCTTTTT---ATACCTACTTCA---ATGGAG---TGG---GAG------CACTTTTATCCA---CGTGCC---GAT---CNGACT---TACGCT---GAAATTCCA
Orconectes_chickasawae_AY701216_CO1like   AAAAAGATTTAGAT!TCTTTTT---ATACCTACTTCA---ATANAA---TGA---CAG------CATTTTTACCCA---CCTGCT---GAT---CACACT---TACTCT---GAAATTCCA
Orconectes_compressus_AY701217_CO1like    TAAAAGATTTAGGTCCTNTTTY!!TATACCTACTTCA---ATAGAA!!GTGR---CNR------CATTCTTTTCCY---CCTGCT---GAT---CACACC!!TTACTCT---GAAATTCCA
Orconectes_macrus_AY701236_CO1like        AAAGAGGTTTGGATCTCTTTTT---ATGCCTACTTCA---ATAGAA---TGG---CAN------CATTCTTTTCCA---CCTGCT!!GGAC---CACACT---TACGCT---GAAATTCCA
Orconectes_australis_AY701204_CO1like     AAAGAGGTANNNATCTCTTTTT---ATA!CTANTTCG---GNGGAA---TGA---CAG------CATTCCTTCCCT---CCTGCT---GATCCNCAC!!T---TACNCT---GAAATTCCA
Orconectes_holti_AY701225_CO1like         ANAAAGATTTANATCTCTTTTT---ATACCTACTTCA---ATANAA---TGA---CAN------CATTTTTACCCC!NCCCTGCG---GAT---CACACT---TACNCT---GAAATTCCA
Orconectes_indianensis_AY701198_CO1like   AAAARRNTTNAGTTCTCTTTTT---ATACCTACTTCA---ATAGAG---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACGCT---GAAATTCCA
Orconectes_forceps_AY701231_CO1like       RAAARNYTYNRGNCNNYTTTTN---NNNCYNNYTYNA---ATANAA---TGG---CAG------CATTCTTTTCCA---CCTGCT---GAT---CACACT---TACTCT---GAAATTCCA
```

Figure 5 Excerpt of the crayfish *COI* nucleotide data set of Taylor and Knouft (2006) aligned by MACSE and visualized by SeaView using codon colors. Note the numerous frameshifts ('!') inferred by MACSE in the bad quality "COI-like" sequences.

MACSE inferred 20 sequences containing frameshifts including 18 annotated as "COI-like", but also two other sequences, *Orconectes margorectus* (AF474362) and *O. propinquus* (AF474357), containing frameshifts caused by additional nucleotides. Frameshifts in those sequences have probably gone unnoticed because they both occur close to the end of the sequences and do not lead to stop codons in the few nucleotides following the insertions. MACSE allows automatically spotting such cryptic cases. The worst "COI-like" sequence was *O. australis* (AY701204) that contains four frameshifts and an internal stop codon due to multiple missing and additional nucleotides. For six "COI-like" sequences, MACSE did not infer the presence of any frameshifts or stop codons: *O. inermis* (AY701201), *O. kentuckiensis* (AF474369 and AY701196), *O. meeki meeki* (AY701213), *O. neglectus chaenodactylus* (AY701240) and *O. pellucidus* (AY701203). Careful visual inspection showed that most of these problems are likely stemming from sequencing errors rather than representing signs of pseudogenization (*numts*).

3.1.2 "COI-like" crustacean sequences in GenBank

The second example data set (BT2009) corresponds to the crustacean "COI-like" sequences harvested from GenBank and presented in Table 1 of Buhay (2009). This data set contains a mix of 54 "COI-like" complete and partial sequences. These sequences were manually assessed in great detail by Buhay (2009) including three sequences that were determined to likely not represent genuine *COI* sequences. We used this example data set to evaluate the capacity of MACSE to automatically detect sequencing errors that have been previously assessed by an expert eye. The *trimNonHomologousFragments* subprogram of MACSE allows identifying and trimming non homologous sequence fragments before further alignment. It could be used to identify entirely non-homologous sequences since such sequences will be trimmed along their entire lengths. A visualization of the result of this pre-filtering process can be output in a `.fasta` file (`-out_mask_detail` option) in which the removed nucleotides are written in lowercase letters. The *trimNonHomologousFragments* subprogram also outputs a `.csv` file detailing the statistics of this pre-filtering process on each sequence (`-out_trim_info` option). This file contains the number of nucleotides and the proportion of each sequence that have been removed:

```
$ java -jar ./macse_v2.03.jar
        -prog trimNonHomologousFragments
        -gc_def 5
        -seq BT2009.fasta
        -out_mask_detail BT2009_trim_mask.fasta
        -out_trim_info BT2009_trim_stats.csv
        -out_NT BT2009_trim_NT.fasta
        -out_AA BT2009_trim_AA.fasta
```

This non-homologous fragment-trimming step allowed to automatically detect the three sequences that are likely not true mitochondrial *COI* sequences: *Chthamalus dalli* (AY795367), *Farfantepenaeus subtilis* (AY344198), and *Fenneropenaeus indicus* (AY395245). Indeed, MACSE automatically removed these sequences in their entirety. In addition, six other sequences were found to contain divergent fragments representing up to 30% of their length that were masked in the resulting output: *Scopelocheirus schellenbergi* (AY830432), *Parastacoides tasmanicus* (AF482492), *Orconectes indianensis* (AY701198), *Orconectes forceps* (AY701231), *Liberonautes chaperi* (AF399977), and *Caligus* sp. (EF452643). Most of these sequences were spotted as very low quality sequences by Buhay (2009) with sloppy 5' or 3' ends and lots of ambiguities.

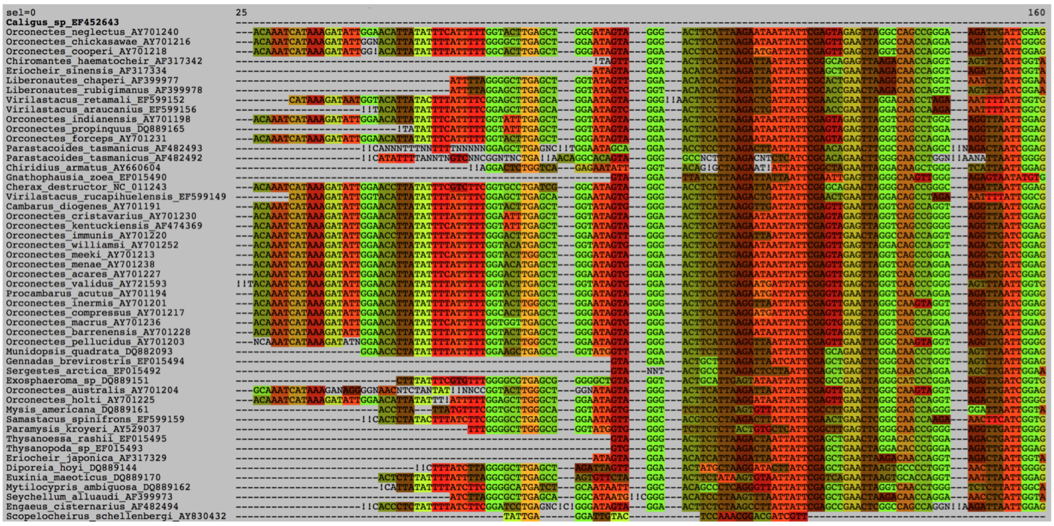
In order to further assess the quality of the remaining 51 "COI-like" sequences, we aligned the masked sequences against the same reference *COI* sequence of *Cherax destructor* (NC_011243) previously used for the TK2006 data set. We used differential costs for frameshifts (`-fs 30`) and stop codons (`-stop 30`) for the reference sequence, and `-fs_lr 10` and `-stop_lr 10` for the less reliable "COI-like" sequences:

```
$ java -jar ./macse_v2.03.jar
        -prog alignSequences
        -gc_def 5
        -seq Cherax_destructor_ref.fasta
        -seq_lr BT2009_trim_NT.fasta
        -fs 30
        -stop 30
        -fs_lr 10
        -stop_lr 10
        -out_NT BT2009_trim_macse_NT.fasta
        -out_AA BT2009_trim_macse_AA.fasta
```

The resulting nucleotide alignment was visually inspected using SeaView (Figure 6).

The statistics on the number of frameshifts and stop codons inferred per sequence in these alignments were computed using the *exportAlignment* subprogram of MACSE:

```
$ java -jar ./macse_v2.03.jar
        -prog exportAlignment
        -gc_def 5
        -align BT2009_trim_macse_NT.fasta
        -out_stat_per_seq BT2009_trim_macse_NT_stat.csv
```



```
sel=0                                 25                                                                                                                                  160
Caligus_sp_EF452643                   ----------------------------------------------------------------------------------------------------------------------------------------
Orconectes_neglectus_AY701240         ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGTACTTGAGCT---GGGATAGTA---GGG---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGCCAGCCGGGA---AGATTGATTGGAG
Orconectes_chickasawae_AY701216       ---ACAAATCATAAAGATATTGGNACATTATATTTTATTTTTGGGGCTTGAGCT---GGAATAGTA---GGG---ACTTCATTAAGAATAATTATTCGAGTTGAATTAGGCCAACCAGGA---AGATTGATCGGGG
Orconectes_cooperi_AY701218           ---ACAAATCATAAAGATATTGG!ACATTATATTTTATTTTTGGCACTTGAGCT---GGGATAGTA---GGA---ACCTCATTAAGGATGATTATTCGAGTTGAGCTAGGTCAACCAGGG---AGATTAATTGGAG
Chiromantes_haematocheir_AF317342     ------------------------------------------------------------!TAGTT---GGT---ACTTCATTAAGATTAATTATTCGGGCAGAGTTAAGACAACCAGGT---AGTTTAATTGGTA
Eriocheir_sinensis_AF317334           ------------------------------------------------------------ATAGTA---GGA---ACATCATTAAGATTAATTATTCGAGCTGAATTAAGACAACCAGGT---AGATTAATTGGTA
Liberonautes_chaperi_AF399977         ------------------------------------ATTTTAGGGGCTTGAGCT---GGTATAGTT---GGA---ACATCACTTAGATTAATTATTCGCACTGAATTAAGGCAACCTGGT---AATCTTATTGGAA
Liberonautes_rubigimanus_AF399978     ------------------------------------ATTTTAGGAGCTTGAGCT---GGTATAGTC---GGT---ACATCACTTAGATTAATTATTCGCACCGAACTAAGACAACCAGGT---AGTTTAATCGGAA
Virilastacus_retamali_EF599152        ---------CATAAAGATAATGGTACATTATACTTTATTTTCGGAGCTTGAGCA---GGAATAGTA---GGG!!AACTTCTTTAAGACTGATTATTCGAACCGAATTAGGACAACCTAGA---AATTTTATTGGTG
Virilastacus_araucanius_EF599156      ---------------------!!TACATTATATTTTATTTTCGGAGCTTGAGCA---GGAATAGTC---GGA---ACTTCCTTAAGACTGATTATCCGAACCGAATTAGGACAACCAAGA---AATTTTATTGGCG
Orconectes_indianensis_AY701198       ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGTATTTGAGCT---GGTATAGTA---GGA---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGTCAGCCTGGG---AGGTTAATTGGAG
Orconectes_propinquus_DQ889165        ---------------------------!TATATTTCATTTTTGGTATTTGAGCT---GGTATAGTA---GGA---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGTCAGCCTGGA---AGGTTAATTGGAG
Orconectes_forceps_AY701231           ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGTACTTGAGCT---GGGATAGTA---GGA---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGGCAGCCAGGA---AGGTTAATTGGAG
Parastacoides_tasmanicus_AF482493     ---------------------!!CANNNTTTNNTTTTNNNNNGGAGCTTGAGNC!!TGGAATAGCA---GGA---ACCTCCTTAAGACTTATTATTCGAACAGAACTTGGCCAACCAGGC!!NAGACTAATCGGAG
Parastacoides_tasmanicus_AF482492     ---------------------!!CATATTTTANNTNGTCNNCGGNTNCTGA!!AACAGGCACAGTA---GGG---GCCNCTTTAAGACNTCTCATCCGCACAGAACTTGGGCAACCTGGN!!AANATTAATTGGGG
Chiridius_armatus_AY660604            ---------------------------------------!!AGGACTCTGGTCA---GAGAATATT---GGT---ACAG!GCTAAGAAT!ATTATTCGGCTAGAATTAGGACAAGCTGGG---TCATTAATTGGAG
Gnathophausia_zoea_EF015490           ---------------------------------------------------------------GTA---GGA---TTATCTTTAAGATTATTAATTCGAATTGAGTTAGGGCAAGTTGGG---AGAGTAATATGTG
Cherax_destructor_NC_011243           ---ACAAATCATAAAGATATTGGAACCTTATATTTCGTCTTCGGTGCCTGATCG---GGCATAGTA---GGC---ACCTCATTAAGATTAATTATCCGGGCAGAACTAGGGCAACCGGGC---AGATTAATTGGAG
Virilastacus_rucapihuelensis_EF599149 ---------CATAAAGATATTGGTACATTATATTTTATTTTCGGAGCTTGAGCA---GGAATAGTG---GGG---ACTTCTTTAAGACTGATTATTCGAACTGAATTAGGACAGCCTAGA---AATTTTATTGGCG
Cambarus_diogenes_AY701191            ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGTACTTGAGCT---GGTATAGTG---GGG---ACTTCATTAAGGTTGATTATTCGAGTTGAATTAGGTCAGCCAGGT---AGGTTAATTGGAG
Orconectes_cristavarius_AY701230      ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGAATTTGAGCT---GGTATAGTA---GGA---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGTCAGCCGGGG---AGGTTAATTGGAG
Orconectes_kentuckiensis_AF474369     ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGTATTTGAGCT---GGTATAGTA---GGA---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGTCAGCCGGGG---AGGTTAATTGGAG
Orconectes_immunis_AY701220           ---ACAAATCATAAAGATATTGGAACCTTATATTTTATTTTTGGGACTTGAGCT---GGAATAGTT---GGG---ACTTCGTTAAGATTAATTATTCGAGTTGAACTAGGCCAACCAGGT---AGGTTGATCGGAG
Orconectes_williamsi_AY701252         ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGTACATGAGCT---GGTATAGTG---GGG---ACTTCATTAAGAATAATTATTCGAGTTGAGTTAGGACAGCCAGGT---AGACTGATCGGAG
Orconectes_meeki_AY701213             ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGTACTTGGGCT---GGGATAGTA---GGA---ACTTCATTGAGAATAATTATTCGAGTAGAGTTAGGTCAGCCGGGA---AGGCTGATCGGAG
Orconectes_menae_AY701238             ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGAACATGAGCT---GGGATAGTG---GGA---ACTTCATTAAGAATAATTATTCGAGTTGAGTTAGGTCAGCCAGGA---AGATTGATTGGAG
Orconectes_acares_AY701227            ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGGACTTGAGCT---GGAATAGTG---GGA---ACTTCATTGAGAATAATTATTCGAGTTGAGTTAGGTCAACCAGGG---AGATTGATCGGAG
Orconectes_validus_AY721593           !!TACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGTACTTGAGCT---GGTATAGTG---GGA---ACTTCATTGAGAATGATTATTCGGGTTGAGCTGGGTCAACCCGGG---AGTTTAATTGGTG
Procambarus_acutus_AY701194           ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGTACTTGAGCT---GGGATAGTA---GGG---ACTTCATTAAGAATGATTATTCGGGTAGAATTAGGTCAACCAGGA---AGATTAATTGGGG
Orconectes_inermis_AY701201           ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGTACTTGGGCT---GGAATAGTA---GGG---ACTTCATTAAGGTTAATTATTCGAGTTGAATTAGGCCAAGTAGGT---AGGTTGATCGGAG
Orconectes_compressus_AY701217        ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGCACTTGAGCT---GGGATAGTA---GGA---ACTTCATTAAGGATGATTATCCGAGTTGAGCTAGGTCAACCAGGG---AGATTAATTGGGG
Orconectes_macrus_AY701236            ---ACAAATCATAAAGATATTGGAACATTATATTTCATTTTTGGTGGTTGAGCC---GGGATAGTG---GGG---ACTTCATTAAGAATAATTATTCGGGTGGAGCTAGGTCAGCCGGGA---AGGCTAATTGGAG
Orconectes_barrenensis_AY701228       ---ACAAATCATAAAGATATTGGAACATTATATTTTATTTTTGGTACTTGAGCT---GGGATAGTA---GGA---ACTTCATTAAGAATAATTATTCGAGTAGAGTTAGGGCAGCCAGGA---AGACTAATTGGAG
Orconectes_pellucidus_AY701203        ---NCAAATCATAAAGATATNGGAACATTATATTTTATTTTTGGTACTTGGGCT---GGAATAGTA---GGG---ACTTCATTAAGGTTAATTATTCGAGTTGAATTAGGCCAAGTAGGT---AGGTTGATTGGGG
Munidopsis_quadrata_DQ882093          ---------------------GGAACCCTATATTTTATTTTTGGAAGCTGAGCC---GGTATGGTT---GGA---ACTTCTTTAAGATTAATTATTCGTGCAGAACTTGGACAACCGGGG---AGTTTAATTGGAG
Gennadas_brevirostris_EF015494        ---------------------------------------------------------------GTA---GGA---ACTGCTTTAAGATTAATCATTCGAGCTGAACTCGGGCAACCTGGT---AGTCTTATTGGAG
Sergestes_arctica_EF015492            ---------------------------------------------------------------GTA---NNT---ACTGCCTTAAGACTCCTAATTCGAGCTGAGCTTGGACAACCAGGA---AGCTTGATTGGAA
Exosphaeroma_sp_DQ889151              ---------------------------CTTTATTTCGTGTTTGGGGCGTGAGCG---GGGGCTGTA---GGT---ACTGCATTGAGTATAATTATTCGCGCCGAACTAGGGCATCCCGGA---AGGTTGATCGGTG
Orconectes_australis_AY701204         ---GCAAATCATAAAGANAGGGGNAACNTCTANTAT!!NNCCGGTACTTGGGCT---GGNATAGTA---GGG---ACTTCATTAAGATTAATTATTCGAGTTGAATTGGGCCAAGTAGGT---AGATTGATCGGAG
Orconectes_holti_AY701225             ---ACAAATCATAAAGATATTGGAACATTATATTT!ATTTTTGGAGCTTGGGCT---GGAATAGTA---GGG---ACTTCATTAAGGATAATTATTCGAGTTGAACTAGGCCAACCAGGT---AGACTAATCGGGG
Mysis_americana_DQ889161              ------------------------ACCTTA---TTATGTTTCGGTGCTTGGGCA---GGTATAGTG---GGT---TCTTCATTAAGTGTTATTATTCGCCTTGAACTTGGGCAGCCAGGG---GGATTAATTGGTA
Samastacus_spinifrons_EF599159        ---------------------!!CACTCTATACTTTATCTTCGGGGCCTGAGCG---GGAATAGTT---GGC---ACGTCCCTAAGACTTATTATTCGAACTGAACTCGGCCAACCAAGA---AACTTCATCGGTG
Paramysis_kroyeri_AY529037            ---------------------------------------TTTGGGGCTTGGGCG---GGTATGGTG---GGG---TCTTCTCTTACTGTGCTCATTCGGCTTGAGTTGGGGCAACCGGGG---AGGTTGATTGGGG
Thysanoessa_rashii_EF015495           ---------------------------------------------------------------GTA---GGG---ACTTCTTTAAGACTAATTATTCGAGCTGAACTAGGACACCCTGGA---AGACTGATTGGAG
Thysanopoda_sp_EF015493               ---------------------------------------------------------------GTG---GGT---ACTTCTCTAAGACTAATTATTCGAGCTGAACTAGGACACCCGGGA---AGATTGATTGGAG
Eriocheir_japonica_AF317329           ------------------------------------------------------------ATAGTA---GGA---ACATCATTAAGATTAATTATTCGAGCTGAATTAAGACAACCAGGT---AGACTAATTGGTA
Diporeia_hoyi_DQ889144                ------------------------------!!CTTTATCTTAGGGGCTTGAGCT---AGATTAGTT---GGA---ACTATGCTAAGGATACTTATCCGAGCTGAATTAAGTGCCCCTGGT---AACTTAATTGGGG
Euxinia_maeoticus_DQ889170            ------------------------ACTCTTTATTTTATTTTAGGCGCTTGAGCC---AGTGTTCTA---GGA---ACTTCTATAAGTGTAATTATTCGGTCAGAATTAAGTGCACCAGGT---AATTTAATTGGAG
Mytilocypris_ambiguosa_DQ889162       ------------------------!CATTATATTTTATCTTCGGGGCATGATCT---GCAATAATT---GGC---ACAGCCCTCAGGGTTATTATTCGAGCTGAATTAGGTCAACCTGGG---TCTCTTATTGGCA
Seychellum_alluaudi_AF399973          ------------------------------------ATCTTAGGCGCTTGAGCA---GGCATAATG!!CGGG---ACATCCTTAAGTTTAATTATTCGCGCTGAACTAAGACAACCAGGA---AACCTCATTGGTA
Engaeus_cisternarius_AF482494         ---------------------!!CACCCTCTATTTTATCTTCGGATCCTGAGNC!C!GGGATAGTA---GGA---ACTTCTCTAAGCCTTATTATTCGAGCCGAACTCGGCCAGCCAGGN!!AAGATTAATTGGAG
Scopelocheirus_schellenbergi_AY830432 ---------------------------------------------TATTGA------GGATTGTAC------------TCCAAACGGACGATCGTT----------------------------------------
```

Figure 6 Excerpt of the crustacean “COI-like” data set assembled from GenBank by Buhay (2009) trimmed and aligned by MACSE, and visualized by SeaView using codon colors. Note the numerous frameshifts (‘!’) inferred by MACSE in the bad quality sequences.

In all cases but three, MACSE correctly retrieved the problematic frameshifts and stop codons previously identified manually by Buhay (2009). The three exceptions concern sequences containing frameshifts occurring very close from sequence extremities (*Orconectes inermis* AY701201 and *Orconectes macrus* AY701236) and an internal frameshift caused by a 1-bp insertion compensated by a 1-bp deletion few nucleotides downstream (*Orconectes meeki* AY701213).

3.2 MACSE alleviates the impact of *numts* in COI-based barcoding species delimitation

Grasshoppers and crayfish are two invertebrate groups that are well known for presenting a high rate of nuclear integration of mitochondrial DNA (mtDNA) that gives rise to nuclear mitochondrial pseudogenes (*numts*). These *numts* are frequently co-amplified with the PCR targeted mtDNA sequences and could strongly impact barcoding analyses based on the mitochondrial *COI* marker. Song et al. (2008) studied the effect of *numts* on COI-based phylogenetic and barcoding analyses. They used PCR with universal arthropods barcoding primers to amplify *COI* in grasshoppers and crayfish. They showed that universal barcode PCR primers co-amplified numerous *COI numts* in both groups resulting in large overestimates of the number of species that can be delineated by *COI* barcoding. Their grasshopper data set (SG2008) contains 95 *COI* sequences including 88 co-amplified *numts* and their crayfish data set (SC2008) comprises 183 *COI* sequences with 101 co-amplified *numts*. *Numts* accumulate frameshifts and stop codons because they become nonfunctional after nuclear integration and are no longer under selective pressure to conserve the open reading frame. We used these two data sets to illustrate the efficiency of MACSE to identify *numts* based on the presence of frameshifts and stop codons as an automatic quality control step aimed at removing these pseudogene sequences before conducting downstream species delimitation analyses. The SG2008 and SC2008 data sets were aligned using the *alignSequences* subprogram of MACSE without *a priori* specification of potentially less reliable *numts* sequences and with default frameshift (`-fs 30`) and stop codon (`-stop 30`) costs for all sequences:

Grasshoppers:

```
$ java -jar ./macse_v2.03.jar
          -prog alignSequences
          -gc_def 5
          -seq SG2008.fasta
          -fs 30
          -stop 30
          -out_NT SG2008_macse_NT.fasta
          -out_AA SG2008_macse_AA.fasta
```

Crayfish:

```
$ java -jar ./macse_v2.03.jar
          -prog alignSequences
          -gc_def 5
          -seq SC2008.fasta
          -fs 30
          -stop 30
          -out_NT SC2008_macse_NT.fasta
          -out_AA SC2008_macse_AA.fasta
```

The statistics on the number of frameshifts and stop codons inferred for each sequence in the resulting alignments were computed using the *exportAlignment* subprogram of MACSE:

Grasshoppers:

```
$ java -jar ./macse_v2.03.jar
          -prog exportAlignment
          -gc_def 5
          -align SG2008_macse_NT.fasta
          -out_stat_per_seq SG2008_macse_NT_stat.csv
```

Crayfish:

```
$ java -jar ./macse_v2.03.jar
          -prog exportAlignment
          -gc_def 5
          -align SC2008_macse_NT.fasta
          -out_stat_per_seq SC2008_macse_NT_stat.csv
```

For the grasshoppers data set, MACSE detected 37 out of the 95 sequences containing at least one frameshift and/or stop codon, and 99 out of the 183 sequences for the crayfish data set. The potential *numt* sequences containing at least one frameshift and/or one stop codon were then removed from the resulting nucleotide alignments. Ambiguously aligned and highly incomplete codon sites were excluded using Gblocks (Castresana, 2000) with default relaxed codon parameters for both the complete and reduced MACSE alignments. Maximum likelihood (ML) phylogenetic inference was then conducted on the four resulting data sets with PhyML v3.1 (Guindon et al., 2009) using a GTR+G8 model and SPR branch swapping on a BIONJ starting tree. The ML phylograms obtained for the grasshoppers and crayfish data sets are presented in Figures 7 and 8, respectively.

In order to gauge the impact of automatically removing *numts* using MACSE on species delimitation, we applied the multi-rate Poisson Tree Process model (Kapli et al. 2017) on the four resulting ML phylograms using the mPTP server (`http://mptp.h-its.org`).

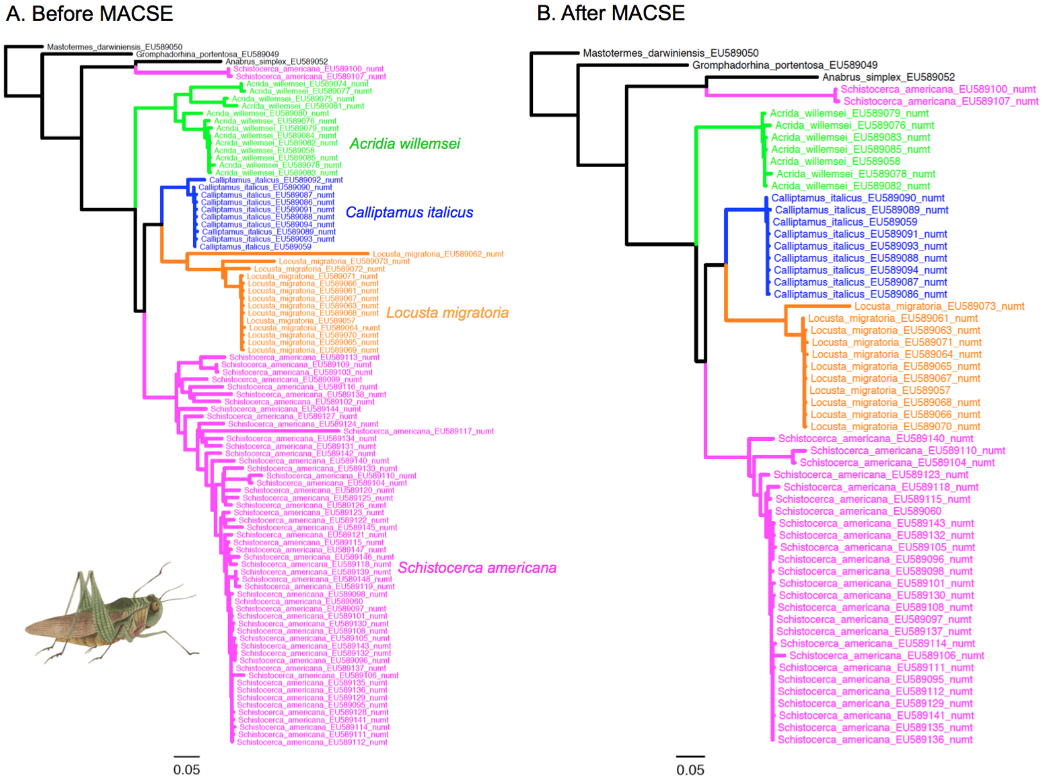

A. Before MACSE
Acridia willemsei
Calliptamus italicus
Locusta migratoria
Schistocerca americana
0.05
B. After MACSE
Mastotermes_darwiniensis_EU589050
Gromphadorhina_portentosa_EU589049
Anabrus_simplex_EU589052
Schistocerca_americana_EU589100_numt
Schistocerca_americana_EU589107_numt
Acrida_willemsei_EU589079_numt
Acrida_willemsei_EU589076_numt
Acrida_willemsei_EU589083_numt
Acrida_willemsei_EU589085_numt
Acrida_willemsei_EU589058
Acrida_willemsei_EU589078_numt
Acrida_willemsei_EU589082_numt
Calliptamus_italicus_EU589090_numt
Calliptamus_italicus_EU589089_numt
Calliptamus_italicus_EU589059
Calliptamus_italicus_EU589091_numt
Calliptamus_italicus_EU589093_numt
Calliptamus_italicus_EU589088_numt
Calliptamus_italicus_EU589094_numt
Calliptamus_italicus_EU589087_numt
Calliptamus_italicus_EU589086_numt
Locusta_migratoria_EU589073_numt
Locusta_migratoria_EU589061_numt
Locusta_migratoria_EU589063_numt
Locusta_migratoria_EU589071_numt
Locusta_migratoria_EU589064_numt
Locusta_migratoria_EU589065_numt
Locusta_migratoria_EU589067_numt
Locusta_migratoria_EU589057
Locusta_migratoria_EU589068_numt
Locusta_migratoria_EU589066_numt
Locusta_migratoria_EU589070_numt
Schistocerca_americana_EU589140_numt
Schistocerca_americana_EU589110_numt
Schistocerca_americana_EU589104_numt
Schistocerca_americana_EU589123_numt
Schistocerca_americana_EU589118_numt
Schistocerca_americana_EU589115_numt
Schistocerca_americana_EU589060
Schistocerca_americana_EU589143_numt
Schistocerca_americana_EU589132_numt
Schistocerca_americana_EU589105_numt
Schistocerca_americana_EU589096_numt
Schistocerca_americana_EU589098_numt
Schistocerca_americana_EU589101_numt
Schistocerca_americana_EU589130_numt
Schistocerca_americana_EU589108_numt
Schistocerca_americana_EU589097_numt
Schistocerca_americana_EU589137_numt
Schistocerca_americana_EU589114_numt
Schistocerca_americana_EU589106_numt
Schistocerca_americana_EU589111_numt
Schistocerca_americana_EU589095_numt
Schistocerca_americana_EU589112_numt
Schistocerca_americana_EU589129_numt
Schistocerca_americana_EU589141_numt
Schistocerca_americana_EU589135_numt
Schistocerca_americana_EU589136_numt
0.05

Figure 7 Maximum likelihood phylograms obtained before (A) and after (B) filtering the grasshoppers data set of Song et al. (2008) using MACSE to detect *numt* sequences containing frameshits and/or stop codons. In this case, MACSE automatically removed 37 *numts*.

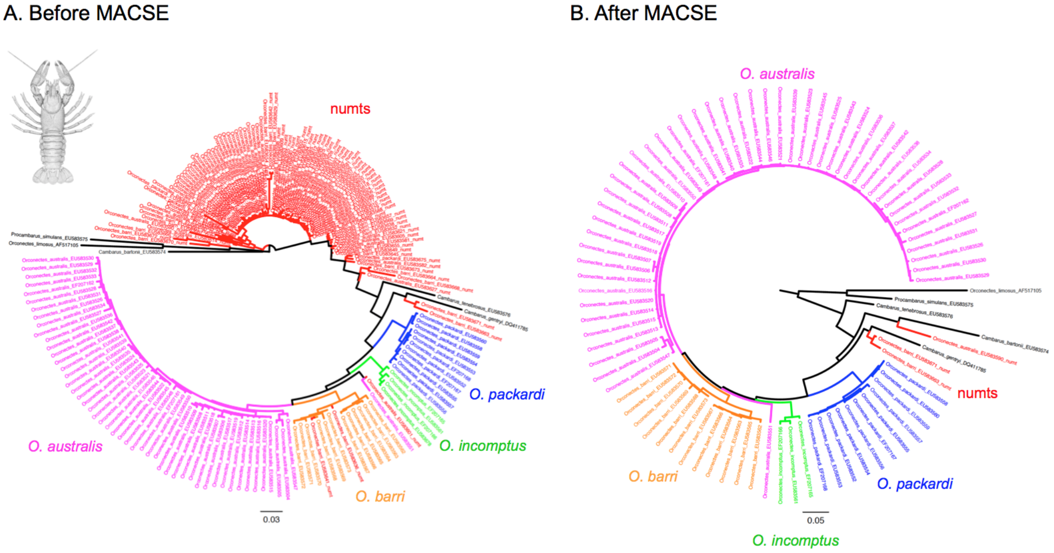


Figure 8 Maximum likelihood phylograms obtained before (A) and after (B) filtering the crayfish data set of Song et al. (2008) using MACSE to detect *numt* sequences containing frameshits and/or stop codons. In this case, MACSE automatically removed 99 *numts* (in red).

For grasshoppers, mPTP delimited 20 species from the 95 sequences of the original data set, which included 88 *numts*. The number of delimited species was halved to 10 with the 58 sequences retained by MACSE that still included 51 potential *numts*, which did not contain any frameshift or stop codon. For crayfish, mPTP delimited 26 species from the 183 sequences of the original data set, which included 101 *numts*. Again, the number of delimited species dropped to 8 with the 84 sequences retained by MACSE that only included 3 potential *numts*, which did not contain any frameshift or stop codon.

3.3 MACSE_BARCODE accurately aligns thousands of metabarcoding sequences

In order to illustrate the efficiency of the MACSE_BARCODE pipeline, we applied the two Nextflow workflows **P__buildRefAlignment** and **P__enrichAlignment** to barcode sequences publicly available for ants, mammals, and flowering plants, which were downloaded through the Taxonomy portal of the BOLD database v4 (`http://v4.boldsystems.org/index.php/TaxBrowser_Home`) on March 3rd 2020 (Table 1).

	BOLD sequences			Homologous sequences		
	per taxa	per taxa and marker	homologous to reference (reverse complemented)	in final alignment	with internal frameshifts	with internal stop codons
Mammalia *COI*	141,145	121,180	117,547 (6)	117,363	223	82
Formicidae *COI*	124,067	121,954	121,792 (33)	121,494	557	16
Magnoliophyta *rbcL*	339,948	121,989	121,598 (116)	121,302	825	346
Magnoliophyta *matK*	339,948	107,413	107,032 (614)	63,250	1,824	143

Table 1 Descriptive statistics of the four BOLD barcoding data sets on which the MACSE_BARCODE pipeline has been applied to construct reference alignments.

3.3.1 Mammalian *COI* sequences in BOLD

As a first example, we aimed at constructing a reference alignment of *COI* barcode sequences for all mammals represented in the BOLD database. As mammalian *COI* sequences are well conserved at the scale of mammals, this first data set serves as an ideal first test case for our approach. Using the Taxonomy portal of the BOLD system v4 (`http://v4.boldsystems.org/index.php/TaxBrowser_Home`), we downloaded the 141,145 publicly available sequences in the Mammalia section. These raw sequences contain sequences from different molecular markers and also include gaps. Sequences corresponding to *COI* can thus be counted using the following command:

```
$ grep -c COI Mammalia_BOLD_141145seq_raw.fasta
```

This resulted in 121,180 *COI* sequences that were extracted and stored in a new fasta file using the following command:

```
$ grep -A1 COI Mammalia_BOLD_141145seq_raw.fasta
   > Mammalia_BOLD_121180seq_COI_raw.fasta
```

At this stage, gaps could be removed from all sequences and illegal characters such as pipes '|' and spaces could be replaced with underscores "_" to ease further bioinformatic processing using for instance:

```
$ sed -e '/>/!s/-//g'
      -e '/>/s/[| :().,;#]/_/g' Mammalia_BOLD_121180seq_COI_raw.fasta
      > Mammalia_BOLD_121180seq_COI.fasta
```

Using the *Homo sapiens* (NC_012920) full length *COI* sequence as a reference sequence, the alignment of representative sequences for the Mammalia *COI* data set could be built using the **P_buildRefAlignment** workflow by running the following command:

```
$ ./nextflow P_buildRefAlignment.nf
        --refSeq Homo_sapiens_NC_012920_COI_ref.fasta
        --seqToAlign Mammalia_BOLD_121180seq_COI.fasta
        --geneticCode 2
        --outPrefix Mammalia_COI
```

This generates a result folder `RESULTS_REFA_Mammalia_COI` containing a `.fasta` file of unaligned representative sequences (`Mammalia_COI_repSeq.fasta`) and the corresponding nucleotide (`Mammalia_COI_final_align_NT.aln`) and amino acid (`Mammalia_COI_final_align_AA.aln`) alignments. A `.fasta` file containing the barcode sequences identified to be homologous to the reference (`Mammalia_COI_homolog.fasta`) and a list of the names of the sequences that have been reverse complemented (`Mammalia_COI_RevComSeqId.list`) are also provided. In this case, 117,547 sequences (97.0%) were found homologous to the *COI* reference sequence and only six sequences had to be reverse complemented to be aligned (Table 1).

The final alignment of all homologous *COI* sequences could then be computed using the **P_enrichAlignment** workflow by providing the previously computed alignment of representative sequences as a reference alignment (`Mammalia_COI_reference_alignment_NT.aln`) and the set of homologous barcode sequences remaining to be aligned (`Mammalia_COI_homolog.fasta`), and executing the following command:

```
$ ./nextflow P_enrichAlignment.nf
        --refAlign Mammalia_COI_final_align_NT.aln
        --seqToAlign Mammalia_COI_homolog.fasta
        --geneticCode 2
        --outPrefix Mammalia_COI
```

This produces a result folder `RESULTS_ENRICH_Mammalia_COI` containing the nucleotide (`Mammalia_COI_alignAll_NT.aln`) and amino acid (`Mammalia_COI_alignAll_AA.aln`) final alignments of all *COI* sequences, as well as versions of those alignments in which frameshifts ('!') have been removed (`Mammalia_COI_alignAll_NT_exp_noFS.aln`) and (`Mammalia_COI_alignAll_AA_exp_noFS.aln`). Moreover, statistics on the steps of sequence trimming (`Mammalia_COI_preTrimingStat.csv`) and alignment enrichment (`Mammalia_COI_enrich_info.csv`) are provided as `.csv` files so that the fate of each initial sequence can be monitored. Here, the final mammal *COI* alignment (Figure 9) contains 117,363 of the initial 121,180 sequences (96.9%). The alignment of 223 of these sequences required the inference of an internal frameshift and 82 sequences have been integrated in this final alignment while having an internal stop codon (Table 1). These sequences could easily have been excluded from the alignment, if needed.

These analyses have been run on a HPC cluster. According to Nextflow, the identification of the representative sequences and the generation of the reference alignment with **P_buildRefAlignment** took only 8 minutes. The obtention of the final alignment with **P_enrichAlignment** required about 134 hours of CPU time but the final result was produced in just 1 hour and 38 minutes thanks to the parallelization used in the pipeline.

The whole MACSE_BARCODE pipeline could also be executed directly using:

```
$ ./nextflow P_macse_barcode.nf
        --refSeq Homo_sapiens_NC_012920_COI_ref.fasta
        --seqToAlign Mammalia_BOLD_121180seq_COI.fasta
        --geneticCode 2
        --outPrefix Mammalia_COI
```

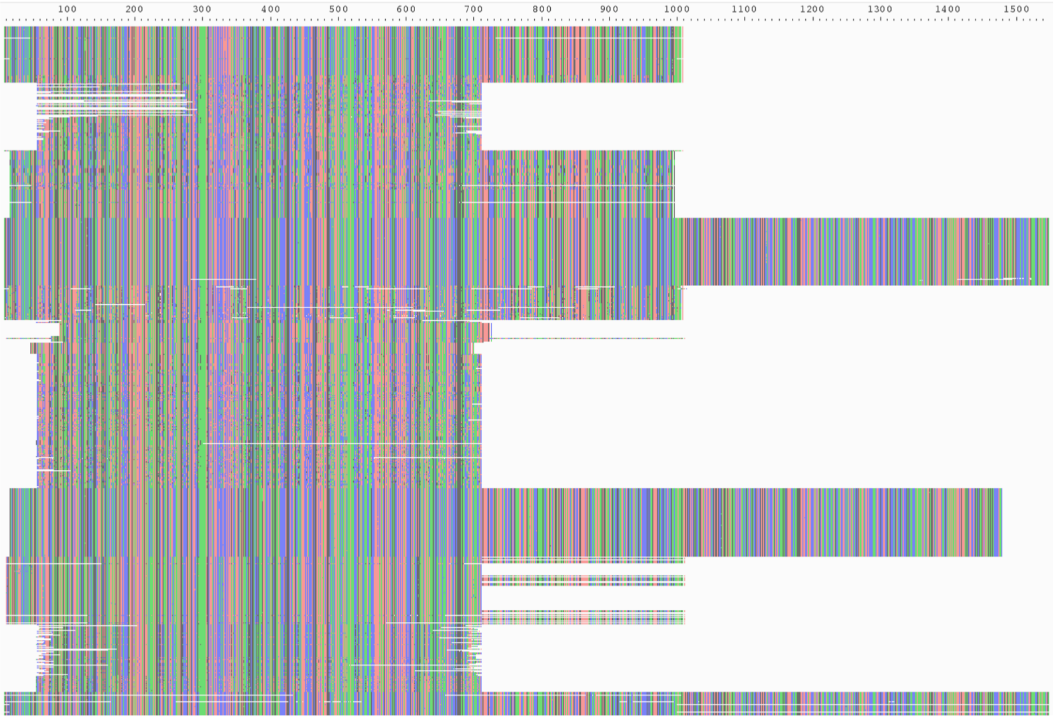


Figure 9 Excerpt of the final Mammalia *COI* nucleotide alignment containing 117,363 sequences produced by MACSE_BARCODE including both full-length (1,548 bp) and shorter *COI* fragments as visualized by AliView (Larsson, 2014).

3.3.2 Ant *COI* sequences in BOLD

As a second example, we considered *COI* barcode sequences from all ant specimens represented in the BOLD database. A total of 124,067 publicly available sequences were downloaded from the Formicidae section using the Taxonomy portal of the BOLD system v4. As for mammals, these raw sequences contain sequences from different molecular markers. So, sequences corresponding to *COI* have to be counted:

```
$ grep -c  COI Formicidae_BOLD_124067seq_raw.fasta
```

The resulting 121,954 *COI* sequences were then extracted and stored in a new `.fasta` file:

```
$ grep -A1  COI Formicidae_BOLD_124067seq_raw.fasta
   > Formicidae_BOLD_121954seq_COI_raw.fasta
```

After gap removal and name cleaning, the alignment of representative sequences for the Formicidae *COI* data set was built with the **P_buildRefAlignment** workflow using the full length *COI* sequence of *Solenopsis geminata* (NC_014669.1) as a reference:

```
$ ./nextflow P_buildRefAlignment.nf
        --refSeq Solenopsis_geminata_NC_014669_COI_ref.fasta
        --seqToAlign Formicidae_BOLD_121954seq_COI.fasta
        --geneticCode 5
        --outPrefix Formicidae_COI
```

In this ant *COI* data set, 121,792 sequences were considered to be homologous to the *COI* reference sequence, and 33 sequences had to be reverse complemented to be aligned (Table 1).

The final alignment of all homologous *COI* sequences was then computed using the **P_enrichAlignment** workflow:

```
$ ./nextflow P_enrichAlignment.nf
        --refAlign Formicidae_COI_final_align_NT.aln
        --seqToAlign Formicidae_COI_homolog.fasta
        --geneticCode 5
        --outPrefix Formicidae_COI
```

The final ant *COI* alignment (Figure 10) comprises 121,494 of the initial 121,954 sequences (99.6%). The alignment of 557 of these sequences required the inference of an internal frameshift and 16 sequences were integrated in this alignment while presenting an internal stop codon (Table 1).

3.3.3 Flowering plant *rbcL* sequences in BOLD

In this third example, we considered another taxonomic group and barcoding marker with the chloroplastic *rbcL* gene, which is the first official barcoding marker for flowering plants. The Taxonomy section of the BOLD system v4 public database contains 339,948 raw public sequences for Magnoliophyta. This included *rbcL* sequences but also other barcoding markers such as the chloroplastic *matK* gene. After counting *rbcL* sequences:

```
$ grep -c  rbcL Magnoliophyta_BOLD_339948seq_raw.fasta
```

The resulting 121,989 *rbcL* sequences were extracted and stored in a new `.fasta` file:

```
$ grep -A1 rbcL Magnoliophyta_BOLD_339948seq_raw.fasta
   > Magnoliophyta_BOLD_121989seq_rbcL_raw.fasta
```

After gap removal and name cleaning, the reference alignment of representative sequences for the Magnoliophyta *rbcL* data set was built with the **P_buildRefAlignment** workflow using the *Magnolia officinalis* (NC_020316.1) full length *rbcL* sequence as a reference:

```
$ ./nextflow P_buildRefAlignment.nf
        --refSeq Magnolia_officinalis_NC_020316.1_rbcL_ref.fasta
        --seqToAlign Magnoliophyta_BOLD_121989seq_rbcL.fasta
        --geneticCode 11
        --outPrefix Magnoliophyta_rbcL
```

In this flowering plant *rbcL* data set, 121,598 sequences were found homologous to the *rbcL* reference sequence among which 116 sequences had to be reverse complemented to be aligned (Table 1). The final alignment of all homologous *rbcL* sequences was then computed using the **P_enrichAlignment** workflow:

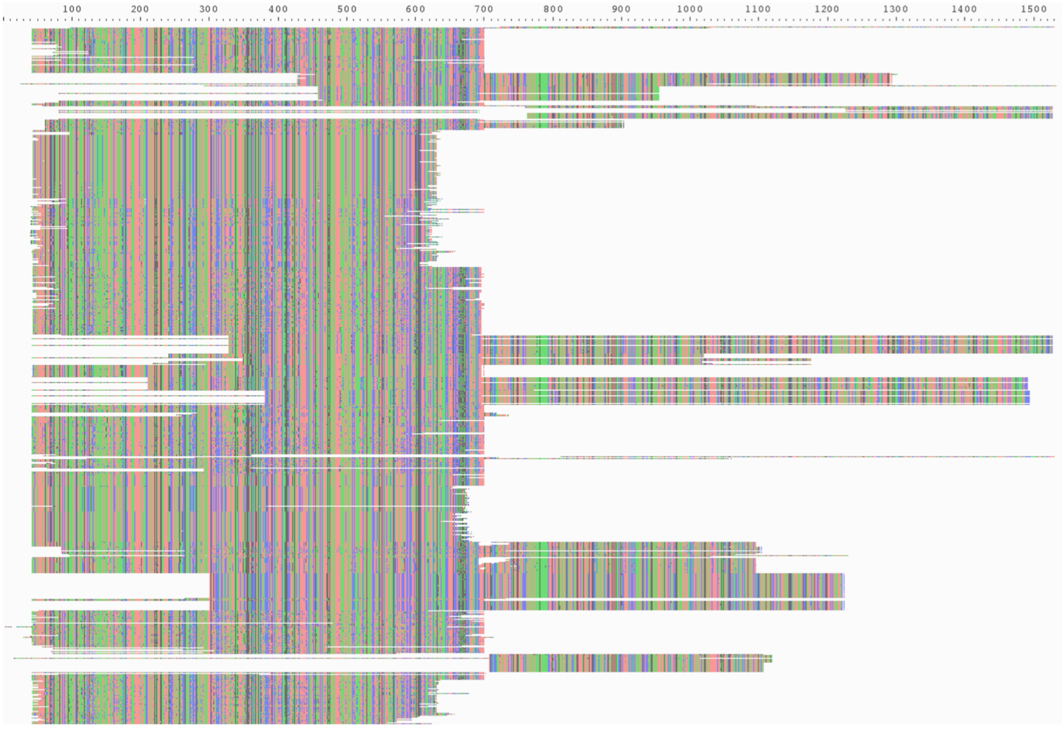


Figure 10 Excerpt of the final Formicidae *COI* metabarcoding nucleotide alignment containing 121,494 sequences produced by MACSE_BARCODE including both full-length (1,533 bp) and shorter *COI* fragments as visualized by AliView.

```
$ ./nextflow P_enrichAlignment.nf
        --refAlign Magnoliophyta_rbcL_final_align_NT.aln
        --seqToAlign Magnoliophyta_rbcL_homolog.fasta
        --geneticCode 11
        --outPrefix Magnoliophyta_rbcL
```

The final flowering plant *rbcL* alignment (Figure 11) comprises 121,302 of the initial 121,989 sequences (99.4%). The alignment of 857 of these sequences required the inference of an internal frameshift and 346 sequences were integrated in this alignment despite the presence of an internal stop codon (Table 1). While this *rbcL* alignment has almost the same number of sequences (121,302) as the alignments obtained for mammals (117,363) and ants (121,494) using *COI*, it contains much more sequences comporting an internal frameshift (825 *versus* 223 and 557, respectively) or an internal stop codon (346 *versus* 82 and 16, respectively). This likely indicates that the sequences available for *rbcL* are of lower quality than those available for *COI*. The high number of sequences containing a stop codon is especially surprising as this should be something easy to check before including a sequence in the BOLD database. For instance, the amino acid sequence displayed on BOLD in the detailed record for the sequence GBVC3450-11, which is flagged as mined from GenBank, includes a stop codon without explicit warning (`http://www.boldsystems.org/index.php/Public_RecordView?processid=GBVC3450-11`).

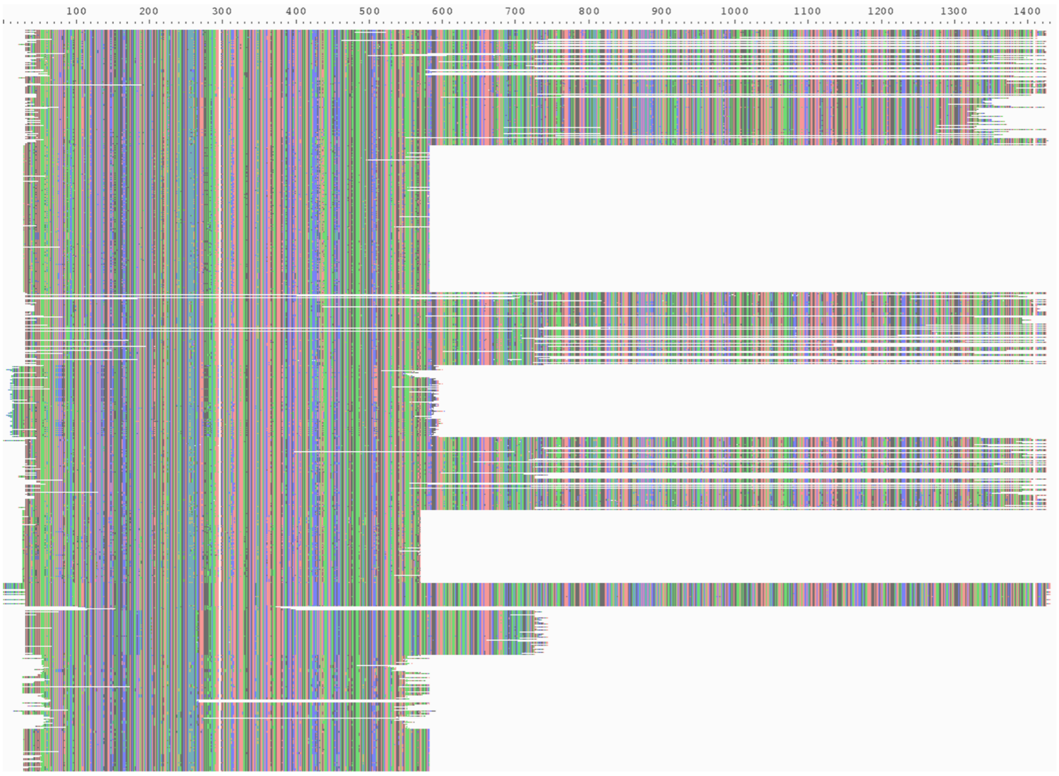


Figure 11 Excerpt of the final Magnoliophyta *rbcL* metabarcoding nucleotide data set containing 121,302 sequences produced by MACSE_BARCODE including both full-length (1,440 bp) and shorter *rbcL* fragments as visualized by AliView.

3.3.4 Flowering plant *matK* sequences in BOLD

For this last example, we considered the second official barcoding marker for flowering plants with the chloroplastic *matK* gene. The *matK* sequences were counted from the previously downloaded raw Magnoliophyta sequences from BOLD:

```
$ grep -c matK Magnoliophyta_BOLD_339948seq_raw.fasta
```

The resulting 107,413 *matK* sequences were extracted and stored in a new `.fasta` file:

```
$ grep -A1 matK Magnoliophyta_BOLD_339948seq_raw.fasta
   > Magnoliophyta_BOLD_107413seq_matk_raw.fasta
```

After gap removal and name cleaning, the reference alignment of representative sequences for the Magnoliophyta *matK* data set was built with the **P__buildRefAlignment** workflow using the *Magnolia officinalis* (NC__020316.1) full length *matK* sequence as a reference:

```
$ ./nextflow P_buildRefAlignment.nf
    --refSeq Magnolia_officinalis_NC_020316.1_matK_ref.fasta
    --seqToAlign Magnoliophyta_BOLD_107413seq_matK.fasta
    --geneticCode 11
    --outPrefix Magnoliophyta_matK
```

In this flowering plant *matK* data set, 107,032 sequences were found homologous to the reference sequence, 614 of which had to be reverse complemented to be aligned (Table 1).

The proportion of sequences provided in the wrong orientation (0.5%) was much higher than for the other data sets (e.g. 0.005% for the mammal *COI* data set).

The final alignment of all homologous *matK* sequences was then computed using the **P_enrichAlignment** workflow:

```
$ ./nextflow P_enrichAlignment.nf
    --refAlign Magnoliophyta_matK_final_align_NT.aln
    --seqToAlign Magnoliophyta_matK_homolog.fasta
    --geneticCode 11
    --outPrefix Magnoliophyta_matK
```

The final flowering plant *matK* alignment (Figure 12) comprises only 63,250 of the initial 107,413 sequences (58.9%). The alignment of 1,824 of these sequences required the inference of an internal frameshift and 143 sequences were aligned while presenting an internal stop codon (Table 1).

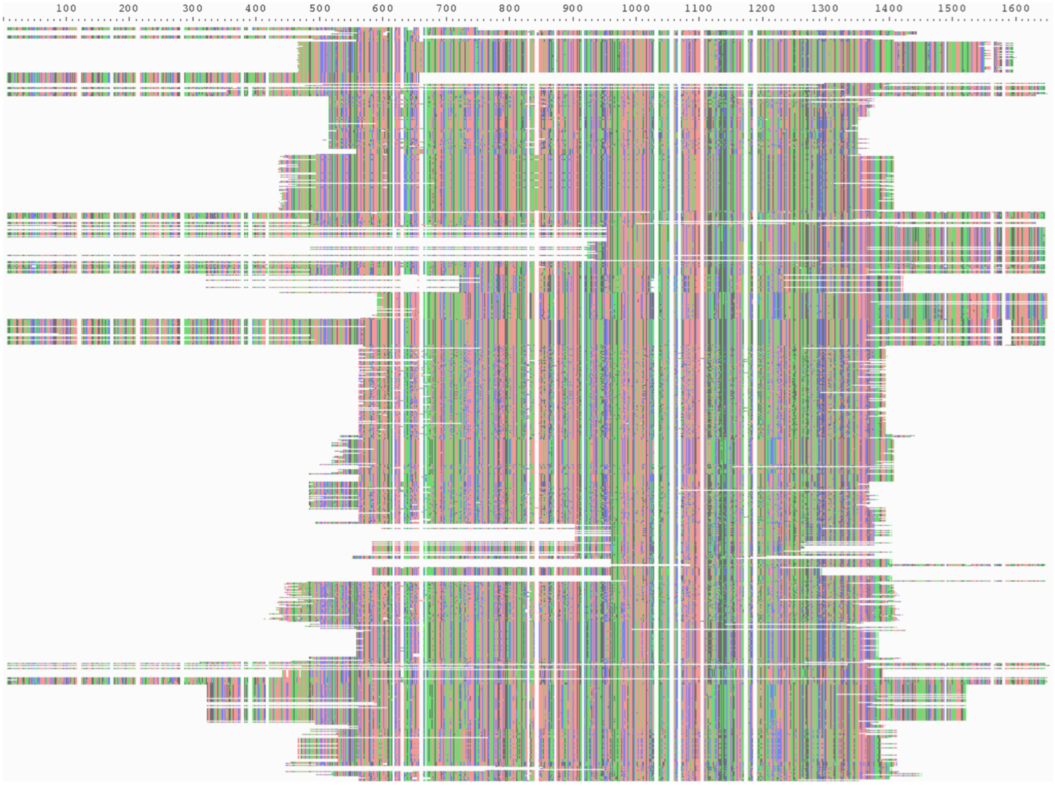


Figure 12 Excerpt of the final Magnoliophyta *matK* metabarcoding nucleotide alignment containing 63,250 sequences produced by MACSE_BARCODE including both full-length (1,536 bp) and shorter *matK* fragments as visualized by AliView.

The fact that more than 40% of the initial *matK* sequences were excluded from the final alignment could reflect the limit of our approach, the poor quality of the sequences available in BOLD for this marker, or more likely a mix of both causes. In fact, many sequences were not inserted because their presence would induce many insertions relative to the reference alignment. This illustrates one limit of our current approach that is based on the conservation of the reference sequence in terms of both amino acid divergence and indel occurrence. Indeed, *matK* is much more variable than *rbcL*, notably in terms of indels (CBOL Plant Working

Group, 2009). However, this could be alleviated by applying the MACSE_BARCODE pipeline at lower taxonomic levels such as the Family level at which *matK* sequences might be more conserved in length. Meanwhile, it seems that the sequences of this data set are of lower quality comparable to the other three data sets with a much higher proportion of the aligned sequences requiring the inference of at least one internal frameshift to be correctly aligned. To ensure that this was not an error of our pipeline, we extracted from the final *matK* alignment the 1,940 sequences (out of a total of 63,250) that were included despite presenting an internal frameshift or a stop codon. This allowed to confirm that, in most cases, the presence of the inferred frameshifts was accurate and that some stop codons appeared right in the middle of the sequences (Figure 13). Altogether, these observations indicate that numerous flowering plant *matK* sequences in BOLD seem to be of relatively poor quality or might represent interesting cases of biologically relevant shifts in translation reading frame, as recently uncovered in Orchidaceae (Barthet et al., 2015).

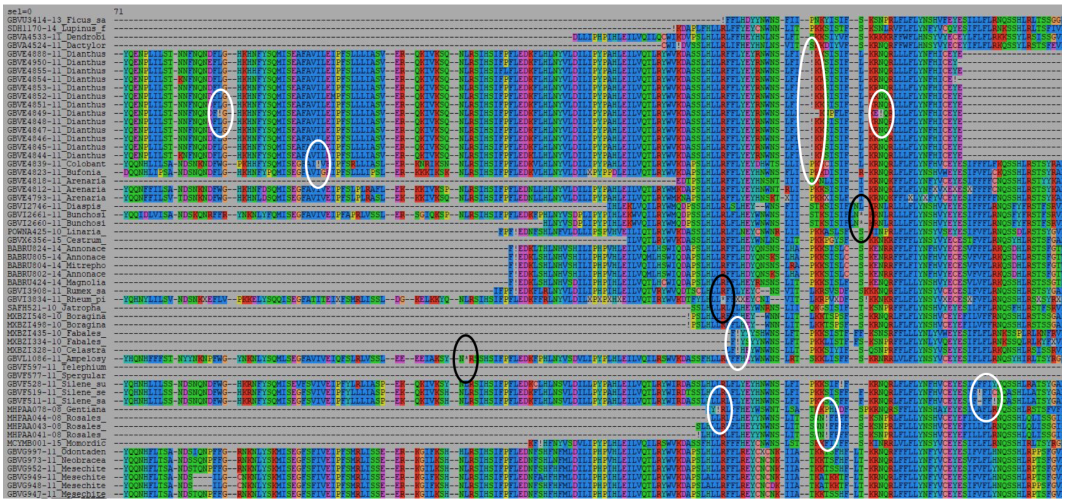


Figure 13 Excerpt of the final Magnoliophyta *matK* final amino acid alignment containing 63,250 sequences focusing on sequences presenting internal frameshifts (white ellipses) and stop codons (black ellipses) as visualized by SeaView.

4 Conclusion

The reference barcoding alignments produced with the MACSE_BARCODE pipeline can be downloaded from the MACSE webpage (`https://bioweb.supagro.inra.fr/macse/index.php?menu=downloadTuto`). Future reference alignments for additional taxonomic groups available in the BOLD database will be distributed through the same webpage. The availability of these quality-controlled alignments for the main protein-coding barcode genes should leverage the power of phylogenetics for taxonomic assignment by allowing to implement probabilistic evolutionary placement in the ever growing range of metabarcoding applications.

Acknowledgements

The project was granted access to the INRA MIGALE bioinformatics platform (`https://migale.inra.fr/`).

References

Andújar, C., Arribas, P., Yu, D. W., Vogler, A. P., and Emerson, B. C. (2018). Why the COI barcode should be the community DNA metabarcode for the metazoa. *Molecular Ecology*, 27(20):3968–3975.

Barthet, M. M., Moukarzel, K., Smith, K. N., Patel, J., and Hilu, K. W. (2015). Alternative translation initiation codons for the plastid maturase MatK: unraveling the pseudogene misconception in the Orchidaceae. *BMC Evolutionary Biology*, 15:210.

Bensasson, D., Zhang, D.-X., Hartl, D. L., and Hewitt, G. M. (2001). Mitochondrial pseudogenes: evolution's misplaced witnesses. *Trends in Ecology and Evolution*, 16(6):314–321.

Berger, S. A., Krompass, D., and Stamatakis, A. (2011). Performance, accuracy, and web server for evolutionary placement of short sequence reads under maximum likelihood. *Systematic Biology*, 60(3):291–302.

Bohmann, K., Evans, A., Gilbert, M. T. P., Carvalho, G. R., Creer, S., Knapp, M., Yu, D. W., and De Bruyn, M. (2014). Environmental DNA for wildlife biology and biodiversity monitoring. *Trends in Ecology and Evolution*, 29(6):358–367.

Boyce, K., Sievers, F., and Higgins, D. G. (2014). Simple chained guide trees give high-quality protein multiple sequence alignments. *Proceedings of the National Academy of Sciences USA*, 111(29):10556–10561.

Buhay, J. E. (2009). "COI-like" sequences are becoming problematic in molecular systematic and DNA barcoding studies. *Journal of Crustacean Biology*, 29(1):96–110.

Calvignac, S., Konecny, L., Malard, F., and Douady, C. J. (2011). Preventing the pollution of mitochondrial datasets with nuclear mitochondrial paralogs (numts). *Mitochondrion*, 11(2):246–254.

Caporaso, J. G., Kuczynski, J., Stombaugh, J., Bittinger, K., Bushman, F. D., Costello, E. K., Fierer, N., Pena, A. G., Goodrich, J. K., Gordon, J. I., et al. (2010). QIIME allows analysis of high-throughput community sequencing data. *Nature Methods*, 7(5):335.

Castresana, J. (2000). Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. *Molecular Biology and Evolution*, 17(4):540–552.

CBOL Plant Working Group (2009). A DNA barcode for land plants. *Proceedings of the National Academy of Sciences USA*, 106(31):12794–12797.

Coissac, E., Riaz, T., and Puillandre, N. (2012). Bioinformatic challenges for DNA metabarcoding of plants and animals. *Molecular Ecology*, 21(8):1834–1847.

DeSantis, T. Z., Hugenholtz, P., Larsen, N., Rojas, M., Brodie, E. L., Keller, K., Huber, T., Dalevi, D., Hu, P., and Andersen, G. L. (2006). Greengenes, a chimera-checked 16S rRNA gene database and workbench compatible with ARB. *Applied and Environmental Microbiology*, 72(7):5069–5072.

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nature Biotechnology*, 35(4):316–319.

Dunning, L. T. and Savolainen, V. (2010). Broad-scale amplification of matK for DNA barcoding plants, a technical note. *Botanical Journal of the Linnean Society*, 164(1):1–9.

Gouy, M., Guindon, S., and Gascuel, O. (2010). SeaView version 4: A multiplatform graphical user interface for sequence alignment and phylogenetic tree building. *Molecular Biology and Evolution*, 27(2):221–224.

Guindon, S., Delsuc, F., Dufayard, J.-F., and Gascuel, O. (2009). Estimating maximum likelihood phylogenies with PhyML. *Methods in Molecular Biology*, 537:113–137.

Hebert, P. D., Cywinska, A., Ball, S. L., and Dewaard, J. R. (2003a). Biological identifications through DNA barcodes. *Proceedings of the Royal Society of London B: Biological Sciences*, 270(1512):313–321.

Hebert, P. D., Ratnasingham, S., and De Waard, J. R. (2003b). Barcoding animal life: cytochrome c oxidase subunit 1 divergences among closely related species. *Proceedings of the Royal Society of London B: Biological Sciences*, 270(suppl_1):S96–S99.

Hebert, P. D., Stoeckle, M. Y., Zemlak, T. S., and Francis, C. M. (2004). Identification of birds through DNA barcodes. *PLoS Biology*, 2(10).

Ji, Y., Ashton, L., Pedley, S. M., Edwards, D. P., Tang, Y., Nakamura, A., Kitching, R., Dolman, P. M., Woodcock, P., Edwards, F. A., et al. (2013). Reliable, verifiable and efficient monitoring of biodiversity via metabarcoding. *Ecology Letters*, 16(10):1245–1257.

Kress, W. J. and Erickson, D. L. (2007). A two-locus global DNA barcode for land plants: the coding rbcL gene complements the non-coding trnH-psbA spacer region. *PLoS One*, 2(6).

Kurtzer, G. M., Sochat, V., and Bauer, M. W. (2017). Singularity: Scientific containers for mobility of compute. *PloS One*, 12(5):e0177459.

Lahaye, R., Van der Bank, M., Bogarin, D., Warner, J., Pupulin, F., Gigot, G., Maurin, O., Duthoit, S., Barraclough, T. G., and Savolainen, V. (2008). DNA barcoding the floras of biodiversity hotspots. *Proceedings of the National Academy of Sciences USA*, 105(8):2923–2928.

Larsson, A. (2014). AliView: a fast and lightweight alignment viewer and editor for large datasets. *Bioinformatics*, 30(22):3276–3278.

Leray, M. and Knowlton, N. (2015). Dna barcoding and metabarcoding of standardized samples reveal patterns of marine benthic diversity. *Proceedings of the National Academy of Sciences USA*, 112(7):2076–2081.

Leray, M., Yang, J. Y., Meyer, C. P., Mills, S. C., Agudelo, N., Ranwez, V., Boehm, J. T., and Machida, R. J. (2013). A new versatile primer set targeting a short fragment of the mitochondrial COI region for metabarcoding metazoan diversity: application for characterizing coral reef fish gut contents. *Frontiers in Zoology*, 10(1):34.

Linard, B., Swenson, K., and Pardi, F. (2019). Rapid alignment-free phylogenetic identification of metagenomic sequences. *Bioinformatics*, 35(18):3303–3312.

Liu, S., Li, Y., Lu, J., Su, X., Tang, M., Zhang, R., Zhou, L., Zhou, C., Yang, Q., Ji, Y., et al. (2013). SOAPBarcode: revealing arthropod biodiversity through assembly of Illumina shotgun sequences of PCR amplicons. *Methods in Ecology and Evolution*, 4(12):1142–1150.

Lopez, J. V., Yuhki, N., Masuda, R., Modi, W., and O'Brien, S. J. (1994). Numt, a recent transfer and tandem amplification of mitochondrial DNA to the nuclear genome of the domestic cat. *Journal of Molecular Evolution*, 39(2):174–190.

Matsen, F. A., Kodner, R. B., and Armbrust, E. V. (2010). pplacer: linear time maximum-likelihood and Bayesian phylogenetic placement of sequences onto a fixed reference tree. *BMC Bioinformatics*, 11(1):538.

Moulton, M. J., Song, H., and Whiting, M. F. (2010). Assessing the effects of primer specificity on eliminating numt coamplification in DNA barcoding: a case study from Orthoptera (Arthropoda: Insecta). *Molecular Ecology Resources*, 10(4):615–627.

Pompanon, F., Deagle, B. E., Symondson, W. O., Brown, D. S., Jarman, S. N., and Taberlet, P. (2012). Who is eating what: diet assessment using next generation sequencing. *Molecular Ecology*, 21(8):1931–1950.

Pruesse, E., Quast, C., Knittel, K., Fuchs, B. M., Ludwig, W., Peplies, J., and Glöckner, F. O. (2007). SILVA: a comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB. *Nucleic Acids Research*, 35(21):7188–7196.

Ramirez-Gonzalez, R., Yu, D. W., Bruce, C., Heavens, D., Caccamo, M., and Emerson, B. C. (2013). PyroClean: denoising pyrosequences from protein-coding amplicons for the recovery of interspecific and intraspecific genetic variation. *PLoS One*, 8(3).

Ranwez, V. and Chantret, N. (2020). Strengths and limits of multiple sequence alignment and filtering methods. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 2.2, pages 2.2:1–2.2:36. No commercial publisher | Authors open access book.

Ranwez, V., Chantret, N., and Delsuc, F. (2020). Aligning protein-coding nucleotide sequences with MACSE. To appear in Methods in Molecular Biology.

Ranwez, V., Douzery, E. J. P., Cambon, C., Chantret, N., and Delsuc, F. (2018). MACSE v2: toolkit for the alignment of coding sequences accounting for frameshifts and stop codons. *Molecular Biology and Evolution*, 35(10):2582–2584.

Ranwez, V., Harispe, S., Delsuc, F., and Douzery, E. J. (2011). MACSE: Multiple Alignment of Coding SEquences accounting for frameshifts and stop codons. *PloS One*, 6(9).

Ratnasingham, S. and Hebert, P. D. (2007). BOLD: The Barcode of Life Data System (http://www. barcodinglife. org). *Molecular Ecology Notes*, 7(3):355–364.

Sarich, V. M. and Wilson, A. C. (1967). Immunological time scale for hominid evolution. *Science*, 158(3805):1200–1203.

Schloss, P. D., Westcott, S. L., Ryabin, T., Hall, J. R., Hartmann, M., Hollister, E. B., Lesniewski, R. A., Oakley, B. B., Parks, D. H., Robinson, C. J., et al. (2009). Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. *Applied and Environmental Microbiology*, 75(23):7537–7541.

Scornavacca, C., Belkhir, K., Lopez, J., Dernat, R., Delsuc, F., Douzery, E. J. P., and Ranwez, V. (2019). OrthoMaM v10: Scaling-up orthologous coding sequence and exon alignments with more than one hundred mammalian genomes. *Molecular Biology and Evolution*, 36(4):861–862.

Singh, T. R., Tsagkogeorga, G., Delsuc, F., Blanquart, S., Shenkar, N., Loya, Y., Douzery, E. J. P., and Huchon, D. (2009). Tunicate mitogenomics and phylogenetics: peculiarities of the Herdmania momus mitochondrial genome and support for the new chordate phylogeny. *BMC Genomics*, 10:534.

Smith, M. A., Fisher, B. L., and Hebert, P. D. (2005). DNA barcoding for effective biodiversity assessment of a hyperdiverse arthropod group: the ants of Madagascar. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 360(1462):1825–1834.

Sochat, V. V., Prybol, C. J., and Kurtzer, G. M. (2017). Enhancing reproducibility in scientific computing: Metrics and registry for Singularity containers. *PloS One*, 12(11):e0188511.

Song, H., Buhay, J. E., Whiting, M. F., and Crandall, K. A. (2008). Many species in one: DNA barcoding overestimates the number of species when nuclear mitochondrial pseudogenes are coamplified. *Proceedings of the National Academy of Sciences USA*, 105(36):13486–13491.

Steinegger, M. and Söding, J. (2017). MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028.

Stoeckle, M. Y. and Kerr, K. C. (2012). Frequency matrix approach demonstrates high sequence quality in avian BARCODEs and highlights cryptic pseudogenes. *PLoS One*, 7(8).

Taberlet, P., Coissac, E., Pompanon, F., Brochmann, C., and Willerslev, E. (2012). Towards next-generation biodiversity assessment using DNA metabarcoding. *Molecular Ecology*, 21(8):2045–2050.

Taylor, C. A. and Knouft, J. H. (2006). Historical influences on genital morphology among sympatric species: gonopod evolution and reproductive isolation in the crayfish genus Orconectes (Cambaridae). *Biological Journal of the Linnean Society*, 89(1):1–12.

Taylor, H. and Harris, W. (2012). An emergent science on the brink of irrelevance: a review of the past 8 years of DNA barcoding. *Molecular Ecology Resources*, 12(3):377–388.

Ward, R. D., Zemlak, T. S., Innes, B. H., Last, P. R., and Hebert, P. D. (2005). Dna barcoding Australia's fish species. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 360(1462):1847–1857.

Woese, C. R. and Fox, G. E. (1977). Phylogenetic structure of the prokaryotic domain: the primary kingdoms. *Proceedings of the National Academy of Sciences USA*, 74(11):5088–5090.

Woese, C. R., Kandler, O., and Wheelis, M. L. (1990). Towards a natural system of organisms: proposal for the domains archaea, bacteria, and eucarya. *Proceedings of the National Academy of Sciences USA*, 87(12):4576–4579.

Yang, C., Wang, X., Miller, J. A., de Blécourt, M., Ji, Y., Yang, C., Harrison, R. D., and Yu, D. W. (2014). Using metabarcoding to ask if easily collected soil and leaf-litter samples can be used as a general biodiversity indicator. *Ecological Indicators*, 46:379–389.

Yu, D. W., Ji, Y., Emerson, B. C., Wang, X., Ye, C., Yang, C., and Ding, Z. (2012). Biodiversity soup: metabarcoding of arthropods for rapid biodiversity assessment and biomonitoring. *Methods in Ecology and Evolution*, 3(4):613–623.

Zuckerkandl, E. and Pauling, L. (1965). Molecules as documents of evolutionary history. *Journal of Theoretical Biology*, 8(2):357–366.