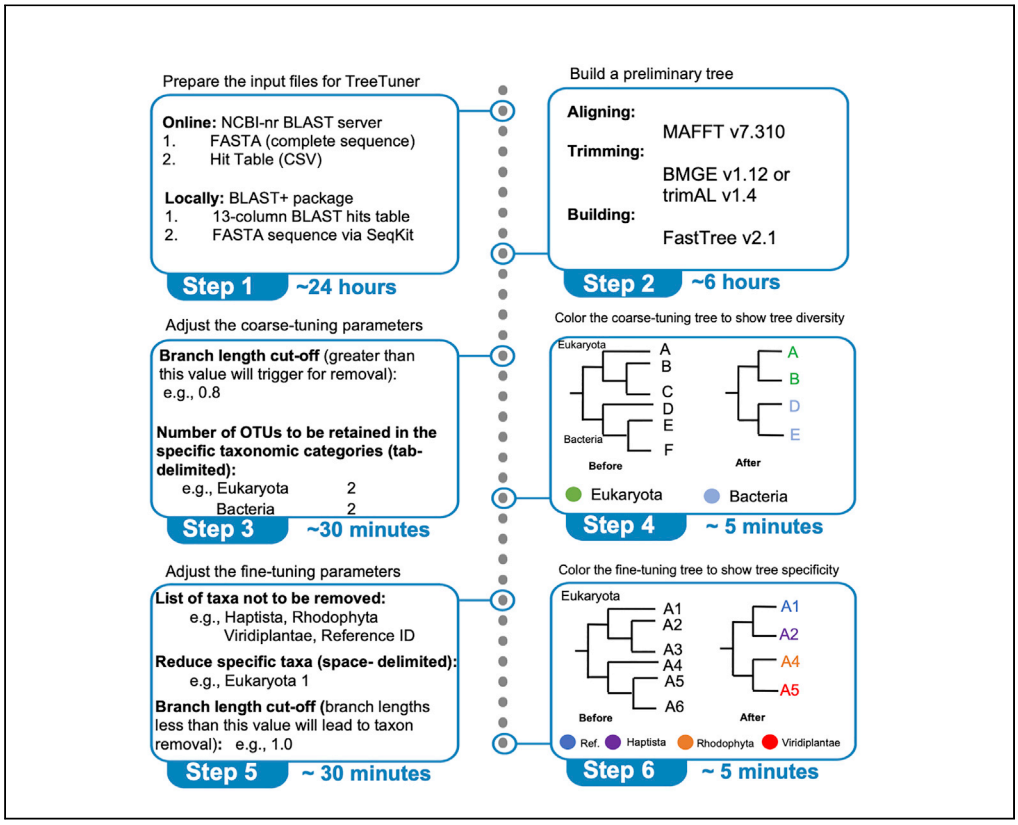
STAR Protocols

CellPress
OPEN ACCESS

Protocol

# TreeTuner: A pipeline for minimizing redundancy and complexity in large phylogenetic datasets




Various bioinformatics protocols have been developed for trimming the number of operational taxonomic units (OTUs) in phylogenetic datasets, but they typically require significant manual intervention. Here we present TreeTuner, a semi-automated pipeline that allows both coarse and fine-scale tuning of large protein sequence phylogenetic datasets via the minimization of OTU redundancy. TreeTuner facilitates preliminary investigation of such datasets as well as more rigorous downstream analysis of specific subsets of OTUs.

Xi Zhang, Yining Hu, Laura Eme, ..., Gina V. Filloramo, Klaas J. van Wijk, John M. Archibald

xi.zhang@dal.ca (X.Z.)
john.archibald@dal.ca (J.M.A.)

## Highlights

Minimizes sequence redundancy in large phylogenetic datasets

Trims thousands of operational taxonomic units (OTUs) from a preliminary tree

Maintains the desired minimal taxonomic diversity and retains specific OTUs for analysis

Available coarse- and fine-tuning options depending on the phylogenetic question at hand

Zhang et al., STAR Protocols 3, 101175
March 18, 2022 
https://doi.org/10.1016/j.xpro.2022.101175

STAR Protocols

CellPress
OPEN ACCESS

Protocol

# TreeTuner: A pipeline for minimizing redundancy and complexity in large phylogenetic datasets

Xi Zhang,[1,2,12,*] Yining Hu,[3] Laura Eme,[1,5] Shinichiro Maruyama,[1,6,7] Robert J.M. Eveleigh,[1,8,9] Bruce A. Curtis,[1,2] Shannon J. Sibbald,[1,2] Julia F. Hopkins,[1,10] Gina V. Filloramo,[1,2] Klaas J. van Wijk,[4,11] and John M. Archibald[1,2,11,13,*]

[1]Department of Biochemistry and Molecular Biology, Dalhousie University, Halifax, NS B3H 4R2, Canada

[2]Institute for Comparative Genomics, Dalhousie University, Halifax, NS B3H 4R2, Canada

[3]Department of Computer Science, Western University, London, ON N6A 5B7, Canada

[4]Department of Plant Biology, Cornell University, Ithaca, NY 14853, USA

[5]Present address: Unité d'Ecologie, Systématique et Evolution, CNRS, Université Paris-Saclay, Paris 91990, France

[6]Present address: Department of Ecological Developmental Adaptability Life Sciences, Graduate School of Life Sciences, Tohoku University, 6-3 Aramaki-aza-Aoba, Aobaku, Sendai 980-8578, Japan

[7]Present address: Graduate School of Humanities and Sciences, Ochanomizu University, Bunkyo-ku, Tokyo 112-8610, Japan

[8]Present address: Canadian Centre for Computational Genomics, McGill University, Montréal, QC H3A 0G4, Canada

[9]Present address: McGill Genome Center, McGill University, Montréal, QC H3A 0G4, Canada

[10]Present address: Foundation Medicine Inc. 150 Second Street, Cambridge, MA 02141, USA

[11]Senior author

[12]Technical contact

[13]Lead contact

*Correspondence: xi.zhang@dal.ca (X.Z.), john.archibald@dal.ca (J.M.A.)
https://doi.org/10.1016/j.xpro.2022.101175

## SUMMARY

**Various bioinformatics protocols have been developed for trimming the number of operational taxonomic units (OTUs) in phylogenetic datasets, but they typically require significant manual intervention. Here we present TreeTuner, a semi-automated pipeline that allows both coarse and fine-scale tuning of large protein sequence phylogenetic datasets via the minimization of OTU redundancy. TreeTuner facilitates preliminary investigation of such datasets as well as more rigorous downstream analysis of specific subsets of OTUs.**

**For complete details on the use and execution of this protocol, please refer to Maruyama et al. (2013) and Sibbald et al. (2019).**

## BEFORE YOU BEGIN

Rapidly expanding genomic databases make it possible to generate sequence alignments with thousands of operational taxonomic units (OTUs) from a wide range of organisms (Yandell and Ence, 2012). Many bioinformatics tools developed for trimming large phylogenetic datasets aim to optimize the diversity of OTUs in a given tree relative to the phylogenetic question at hand (Krishnamoorthy et al., 2011; Menardo et al., 2018; Emms and Kelly, 2021). However, little attention has been given to the development of automated and semi-automated bioinformatic tools capable of both coarse- and fine-grained tuning of OTU complexity in large phylogenetic datasets. Here we describe TreeTuner, a bioinformatics pipeline for researchers wanting to combine the ability to explore species tree diversity with rigorous downstream analysis of specific OTUs. The approach used in TreeTuner has been successfully utilized in earlier analyses of the ubiquitin multigene family in algae and the PsbO protein in the land plant *Arabidopsis thaliana* (Maruyama et al., 2013; Sibbald et al., 2019).

### Data collection and methods overview

We demonstrate the utility of the TreeTuner pipeline using a selected set of ClpS protein coding sequences (i.e., ATP-dependent Clp protease adaptor protein) (Nishimura et al., 2013; Bouchnak and van Wijk, 2019) from the highly curated genome of *A. thaliana* as a query for the retrieval of homologs from the NCBI-nr database. Prior to assembly of a 'master' alignment capturing the full breadth of known sequence diversity, we first build preliminary phylogenies to identify general patterns in the data ('coarse tuning'), and then proceed to curated, in-depth analyses using more sophisticated tree-building methods and optimized taxon sampling ('fine tuning'). The goal of this protocol is to provide a step-by-step guide to the TreeTuner workflow (https://github.com/zx0223winner/TreeTuner) thereby allowing researchers to run the pipeline on their own data.

### Requirements for setting up the pipeline

The TreeTuner pipeline combines the software and tools required for both coarse- and fine-scale tuning. For coarse tuning, to run TreeTrimmer (Maruyama et al., 2013) locally, pre-installed Ruby (e.g., Ruby v2.5.1 and BioRuby v2.0.3) is required. For fine tuning, the necessary custom Perl scripts (*rm_inparal_rank.pl* and *trim2untrim.pl*) and Python script (*rename_ncbi_blastdb.py*) can be found at the following GitHub website: https://github.com/zx0223winner/TreeTuner. Pre-installed Perl (e.g., Perl 5 and BioPerl v1.7.2) is required to run these scripts. To color the OTUs on the Newick tree based on taxonomic rank, the Environment for Tree Exploration (ETE3) toolkit (Huerta-Cepas et al., 2016) and associated Python scripts (e.g., *color_coarse_tuning_tree*.py and *color_fine_tuning_tree.py*) are needed.

TreeTuner users will also need to run their BLAST searches (Altschul et al., 1997) against the NCBI-nr database. Subsequently, the initial raw alignments, trimmed alignments and preliminary tree files can be created by different tools, such as MAFFT v7(Katoh and Standley, 2013), BMGE v1.12 (Criscuolo and Gribaldo, 2010), trimAl v1.4 (Capella-Gutiérrez et al., 2009), FastTree v2.1 (Price et al., 2010) and IQ-TREE v1.6.12 (Nguyen et al., 2015) software packages, which can be accessed via the links in the key resources table. A conda environment definition file (TreeTuner-ENV.yaml) is also provided at the GitHub page to assist users with installing all the required dependencies.

> ***Note:*** At a minimum, a computer with 2 cores, 4 GB of RAM and 256 GB of storage with at least 100 GB of free hard disk space is required; this will allow the OTUs of the phylogenetic tree to be trimmed within a few minutes.

## KEY RESOURCES TABLE

| REAGENT or RESOURCE | SOURCE | IDENTIFIER* |
|---|---|---|
| Deposited data | | |
| *Arabidopsis thaliana* protein sequence (Athaliana_447_Araport11.protein.fa.gz) | Cheng et al. (2017) | JGI Phytozome v13: *Arabidopsis thaliana* Araport11; https://phytozome-next.jgi.doe.gov/info/Athaliana_Araport11 |
| Software and algorithms | | |
| NCBI BLAST v2.2.26 | Altschul et al. (1997) | SCR_004870; ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ |
| NCBI-nr database | Pruitt et al. (2005) | SCR_006472; https://www.ncbi.nlm.nih.gov/refseq/ |
| acc2tax | N/A | GitHub: https://github.com/richardmleggett/acc2tax |
| SeqKit | Shen et al. (2016) | SCR_018926; https://bioinf.shenwei.me/seqkit/ |
| NCBI Taxonomy | Federhen (2012) and Schoch et al. (2020) | SCR_003256; https://ftp.ncbi.nlm.nih.gov/pub/taxonomy/ |
| MAFFT v7.310 | Katoh and Standley (2013) | SCR_011811; https://mafft.cbrc.jp/alignment/server/ |
| BMGE v1.12 | Criscuolo and Gribaldo (2010) | ftp://ftp.pasteur.fr/pub/GenSoft/projects/BMGE/ |

(*Continued on next page*)

*Continued*

| REAGENT or RESOURCE | SOURCE | IDENTIFIER* |
|---|---|---|
| trimAl v1.4 | Capella-Gutiérrez et al. (2009) | SCR_017334; http://trimal.cgenomics.org/downloads |
| FastTree v2.1 | Price et al. (2010) | SCR_015501; http://www.microbesonline.org/fasttree/ |
| IQ-TREE v1.6.12 | Nguyen et al. (2015) | SCR_017254; http://www.iqtree.org |
| TreeTrimmer; Treetrimmer.rb | Maruyama et al. (2013) | https://github.com/zx0223winner/TreeTuner/tree/main/Tutorial/TreeTuner_file_examples/Step11-14_coarse_tuning |
| Conda | N/A | SCR_018317; https://docs.conda.io/en/latest/ |
| ETE3 | Huerta-Cepas et al. (2016) | http://etetoolkit.org |
| Ruby v2.5.1; BioRuby v2.0.3 | The Ruby community | https://www.ruby-lang.org/en/downloads/ |
| Perl 5; BioPerl v1.7.2 | The Perl community | SCR_018313; https://www.perl.org/get.html; https://bioperl.org |
| Python 3 | The Python community | SCR_008394; https://www.python.org/downloads/ |
| pandas v1.2.2 | Python Data Analysis Library | https://pandas.pydata.org |
| rm_inparal_rank.pl; trim2untrim.pl lauralib.pm | This study | https://github.com/zx0223winner/TreeTuner |
| rename_ncbi_blastdb.py; color_coarse_tuning_tree.py; color_fine_tuning_tree.py | This study | https://github.com/zx0223winner/TreeTuner |

*Note:* *Identifier is used from RRID portal (https://scicrunch.org/resources). For instance, the NCBI BLAST has the RRID of SCR_004870.

## MATERIALS AND EQUIPMENT

The TreeTuner pipeline implementation was written in Ruby v2.5.1, Perl 5 and Python 3 using the following software and custom scripts: *treetrimmer.rb*, which enables large phylogenetic datasets to be coarsely trimmed in terms of redundancy. The representative OTUs can be referenced for more rigorous downstream analysis; *rm_inparal_rank.pl* and *trim2untrim.pl* are custom scripts to more finely remove redundancy in the selected OTUs and retain the sequences of interest to build a more robust tree; *rename_ncbi_blastdb.py*, *color_coarse_tuning_tree.py and color_fine_tuning_tree.py* are custom Python scripts to parse the input protein sequences, format the header of NCBI sequence IDs and color the phylogenetic tree files based on taxonomic ranks of the OTUs. A full list of the utilized scripts and database, including links, can be found in the key resources table. The full TreeTuner source code can be found in the GitHub repository. A useful hands-on tutorial (Online_TreeTuner Tutorial.pdf) can also be accessed under the tutorial directory of GitHub.

Our test input data consisted of BLAST search results against the NCBI-nr database, the initial raw alignments, trimmed alignments and preliminary tree files created by different tools from the key resources table. Note that five documents are required before the TreeTuner pipeline can be run. The NCBI-nr database BLAST results are obtained first in a 13-column table (Table 1). The second document contains the query sequences retrieved from NCBI (Figure 1A). The third document is a reference list of OTU names and taxonomic information (Figure 1B), and the fourth document is the untrimmed alignment and trimmed alignment (Figure 1C). The fifth document is the tree in Newick format (Figure 1D). In the following step-by-step protocol, we use the ClpS protein sequence from *A. thaliana* (Cheng et al., 2017) to demonstrate the creation of these files.

## STEP-BY-STEP METHOD DETAILS

### Preparing the online NCBI-nr protein BLAST-search result files

Timing: ~2 min (Depending on file sizes and Internet speed)

For the purposes of demonstration, we select the ClpS protein from *A. thaliana* in this protocol, which is applicable to any other proteins and species. The NCBI-nr protein BLAST-search result

**Table 1. Example of TreeTuner input file based on NCBI-nr database BLAST result**

| Query acc.ver | Subject acc.ver, | % Identity | Alignment length | Mismatches | Gap openings | q. start | q. end, | s. start | s. end | Evalue | Bit score | % Positives |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AT1G68660.1 | NP_564937.1 | 100 | 159 | 0 | 0 | 1 | 159 | 1 | 159 | 3.75E-113 | 327 | 100 |
| AT1G68660.1 | XP_002888676.1 | 97.484 | 159 | 4 | 0 | 1 | 159 | 1 | 159 | 7.51E-111 | 322 | 98.74 |
| AT1G68660.1 | XP_010470825.1 | 93.711 | 159 | 10 | 0 | 1 | 159 | 1 | 159 | 9.73E-108 | 314 | 98.11 |
| AT1G68660.1 | XP_006302940.1 | 93.082 | 159 | 11 | 0 | 1 | 159 | 1 | 159 | 4.01E-107 | 312 | 98.11 |
| AT1G68660.1 | CAA7053525.1 | 94.34 | 159 | 8 | 1 | 1 | 159 | 1 | 158 | 1.46E-106 | 311 | 97.48 |
| AT1G68660.1 | XP_010415494.1 | 93.125 | 160 | 10 | 1 | 1 | 159 | 1 | 160 | 8.87E-106 | 309 | 97.5 |
| AT1G68660.1 | XP_013589864.1 | 93.75 | 160 | 9 | 1 | 1 | 159 | 1 | 160 | N/A | NA/ | N/A |

file is the first document required for the TreeTuner pipeline (Table 1). What follows is a description of how to acquire both online and local BLAST-search results using an example FASTA file (Figure 2A). The sample files can be acquired from GitHub under the tutorial directory. Users will acquire the online BLAST-search results (e.g., FASTA file and Hit Table) in steps 1–2.

1. To run the NCBI BLAST server, users can simply click the link (https://blast.ncbi.nlm.nih.gov/Blast.cgi) and select the protein-to-protein BLASTP option.
   a. Paste in the amino acid sequence (in this case, the ClpS protein from *A. thaliana*).

```
>AT1G68660.1

METAICGRLALAPSSLFNSKSGDKHLVSKGPCVNRSILMTLSTSAALGKGGGVLDKPIIEKTTPGRESEFDLRKSKKIAP

PYRVILHNDNFNKREYVVQVLMKVIPGMTVDNAVNIMQEAHINGLAVVIVCAQADAEQHCMQLRGNGLLSSVEPDGGGC
```

   b. Select the NCBI-nr database (Figure 2A).
   c. Choose the parameters (e.g., the max target sequences: 1000; expect threshold: 1e-10) (Figure 2B).
   d. Click the BLAST button at the bottom.

2. After a few seconds, users can see the BLAST result interface (Figure 2C). Users can then download the FASTA (complete sequence) and Hit Table (CSV) via the Download option tab. The FASTA file will contain the ClpS homologs retrieved from the NCBI-nr database. The Hit Table summarizes the matched hits in a 13-column tabular file (Table 1). These two documents are necessary input files for subsequent steps. Before moving on to the tree building step, an alternate approach (i.e., local BLAST) to acquiring these input files is described at step 3.

### Preparing the local NCBI-nr protein BLAST-search result files

**Timing: ~24 h (Depending on file sizes, computing power, and Internet speed)**

Users can also use the local BLAST approach to acquire the significant hits to the protein query from a customized database. This is especially useful for users to explore the full breadth of homolog diversity, since some of homologs might not be present in, or retrievable from, NCBI-nr. Users will acquire two necessary documents (e.g., BLASTP_clps.tsv and clps_hits.fasta) for the TreeTuner pipeline in steps 3–8.

3. Although it is easy to acquire NCBI-nr protein BLAST results online (step 1–2), we also recommend users perform their BLAST searches locally (steps 3–7 below) for more robust analyses. This is because users can more freely adjust the parameters such as max target sequences. The NCBI BLAST server (https://blast.ncbi.nlm.nih.gov/Blast.cgi) limits the number of retrievable


**Figure 1. Examples of input files for the TreeTuner pipeline**

(A) Query protein sequence retrieved from NCBI.

(B) Reference list of OTU names and taxonomic information.

(C) Example of untrimmed and trimmed files.

(D) The tree in Newick format.

sequences to 5,000, which might not capture all of the sequence diversity in the NCBI-nr database. We recommend choosing a maximum (max) target sequence parameter value between 100,000 and 300,000. A max target sequence parameter of 1,000–5,000 is fine if one wants to quickly test the pipeline, and users can skip steps from 3 to 7 and proceed to the tree building analysis starting at Step 9 directly.

*Note:* Local NCBI-nr protein BLAST allows users to conveniently adjust the parameters of thresholds, such as E-value cut-off and maximum target sequences, and to perform thousands of queries against NCBI-nr and other environment- or taxon-specific databases, such as the Marine Microbial Eukaryote Transcriptome Sequencing Project (MMSTSP) (Keeling et al., 2014; Caron et al., 2017). It should also be noted that there is a need for users to have a basic knowledge of bioinformatics in order to take full advantage of TreeTuner, in particular, the ability to use the basic BLAST package and dash shell in a Linux/Unix environment.

4. To set up the local BLAST, users will need to download the BLAST Tool Package and the FASTA sequences corresponding to the custom database. Also, the "blastdbv5-user-guide.pdf" document in the GitHub "tutorial" directory contains complementary vignettes to help guide users, which has been proved useful in our previously published protocol (Zhang et al., 2021a, 2021b).
   a. Download the BLAST Package via https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/. Please select the appropriate distribution based on your computer operating system (Windows, MacOS or Linux). The default is the Linux version (ncbi-blast-2.12.0+-x64-linux.tar.gz).

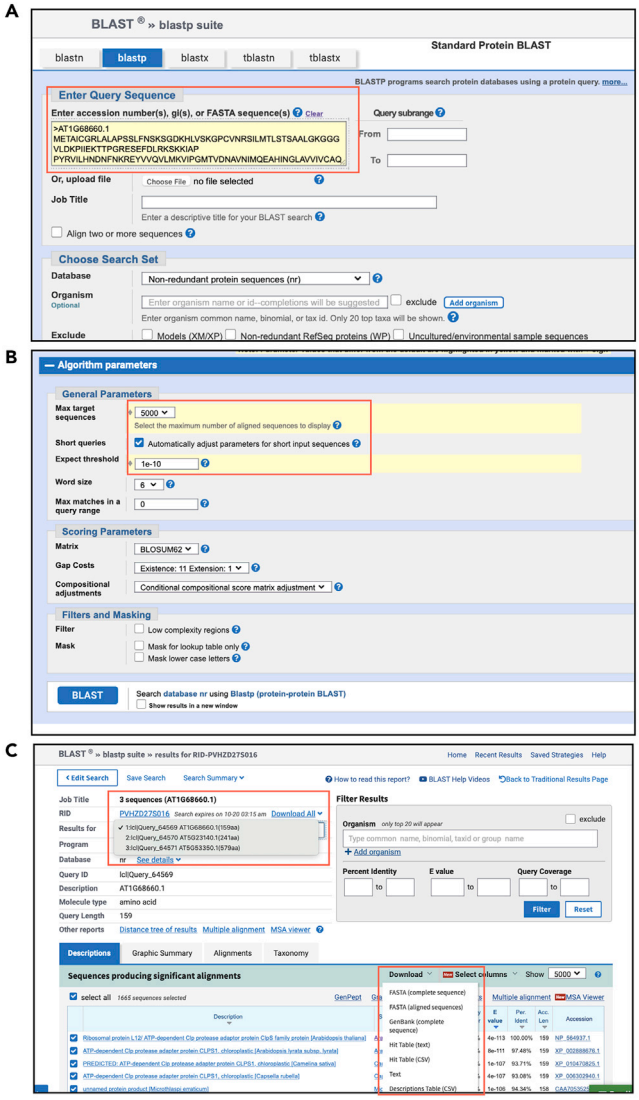



**Figure 2. Screenshots of NCBI online BLAST**

(A) Input file window of the BLAST server.

(B) Parameter adjustments window of Online BLAST.

(C) NCBI BLAST result interface.

b. To simplify the pipeline, we will still use the *A. thaliana* ClpS protein as the example file (clps.fasta).

```
# clps.fasta file

>AT1G68660.1

METAICGRLALAPSSLFNSKSGDKHLVSKGPCVNRSILMTLSTSAALGKGGGVLDKPIIEKTTPGRESEFDLRKSKKIAP

PYRVILHNDNFNKREYVVQVLMKVIPGMTVDNAVNIMQEAHINGLAVVIVCAQADAEQHCMQLRGNGLLSSVEPDGGGC
```

c. Set up a local NCBI-nr database. The NCBI-nr BLAST v5 databases can be accessed via https://ftp.ncbi.nlm.nih.gov/blast/db. Some necessary files (e.g., nr.01.tar.gz, nr.01.tar.gz.md5, and nr.02.tar.gz) can be automatically downloaded via a custom Perl script at step 4e.

d. The makeblastdb command will construct a protein database by taking in the FASTA file with the parameter (-in), setting up the database type (e.g., protein) with the parameter (-dbtype protein), and naming the database (e.g., MMETSP_database) with parameter (-title database_name). The '-out' option will yield the database output name (e.g., MMETSP_db).

```
# Note: makeblastdb file is from the BLAST package and the following command can be run in the terminal.

>./makeblastdb -in MMETSP.fasta -dbtype prot -title MMETSP_prot_database -out MMETSP_db
```

e. To download the NCBI-nr v5 databases, use the Perl script update_blastdb.pl, which is in the downloaded BLAST+ package (https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/). This command will download the NCBI-nr database (https://ftp.ncbi.nlm.nih.gov/blast/db/v5) with the name nr without using the makeblastdb command to recreate the database files. It could take hours for processing, depending on the Internet connection.
   i. Users can first check all available databases via the command below.

```
> perl update_blastdb.pl –blastdb_version 5 –showall

# This will give the results like this:

# Connected to NCBI; downloading BLASTDBv5

# human_genome

# landmark

# ...
```

   ii. Users can then run the command below to automatically download the NCBI-nr database which includes 55 volumes of data (>100 Gb). Or users can manually download these 110 files (i.e., nr.00.tar.gz, nr.00.tar.gz.md5, etc.) from the link: https://ftp.ncbi.nlm.nih.gov/blast/db/v5.

```
> perl update_blastdb.pl –blastdb_version 5 nr –decompress

# This will bring the results like this:

# Connected to NCBI; downloading BLASTDBv5

# Downloading nr (55 volumes) ...

# Downloading nr.00.tar.gz...

# Downloading nr.00.tar.gz.md5

#...
```

5. Use the BLASTP search option to blast the amino acid sequences against NCBI-nr databases. The BLASTP command can carry out the protein similarity search by searching the query file (e.g., clps.fasta) against the protein database using parameters, such as '-evalue' (indicating the significance of the BLAST hits), '-outfmt 6' (meaning the tabular format of the BLAST result), '-out' (telling the file name of the output file; e.g., BLASTP_UWO241_uniprot.xml) and '-max_target_seqs' (potential number of hits).

```
>./blastp -query clps.fasta -db nr -out BLASTP_clps.tsv -evalue 1e-10 -outfmt "6 qseqid sseqid pident length mismatch gapopen qstart qend sstart send evalue bitscore ppos" -max_target_seqs 10000
```

⚠ CRITICAL: The parameters after the option '-outfmt' will yield the desired columns in a tab-delimited output file. For example, 'qseqid' refers to query sequence ID and 'qlen' refers to query length (https://www.metagenomics.wiki/tools/blast/blastn-output-format-6). These desired parameters will create a 13-column table (e.g., Table 1). Make sure to use the amino acid sequences and the BLASTP option, which allows for greater sensitivity compared to BLASTN. The BLAST output parameter must be in tabular format. Users can adjust the parameter of the E-value, but we recommend that it be no greater than 1e-10 (to limit erroneous homology inferences). Troubleshooting 1

6. The local BLAST tabular column is slightly different from the online version. A BLAST-search result example file can be found in the GitHub "tutorial" directory under the name "TreeTuner_file_examples". For example, the gene ID of matched hits from the local BLAST result contain the extra source of ID (e.g., ref|NP_564937.1| compared to NP_564937.1). The BLAST output is in the form of a 13-column tabular file, including the key information from query name to percentage identity.

7. Once users acquire the BLAST hits, they can pull out the FASTA sequences in one of two ways (online or locally).
   a. For example, users can first pull out a list of protein ID from the file (e.g., BLASTP_clps.tsv) like Table 1.

```
> awk -F'\t' '{print $2}' BLASTP_clps.tsv|sed 's/.*\|\(.*\)\|.*/\1/g' > clps_hits_id.txt

# This will yield a list of BLAST hits of ClpS (e.g., NP_564937.1 XP_002888676.1 ...).
```

   b. Users can then use the Batch Entrez (https://www.ncbi.nlm.nih.gov/sites/batchentrez) to retrieve the FASTA sequences corresponding to the list of protein IDs acquired in step 7a.
   c. There is also a way to retrieve these FASTA sequences via the protein ID locally. Users can use the SeqKit (Shen et al., 2016) to quickly pull out the amino acid sequences from the NCBI database.

```
# Prepare a large hard drive to download the nr.gz file (~110 Gb) via https://ftp.ncbi.nlm.nih.gov/blast/db/v5/FASTA/

# Install the latest SeqKit package (e.g., seqkit_linux_386.tar.gz) via https://bioinf.shenwei.me/seqkit/

# Run the seqtk via the commands

>chmod +x seqkit

>seqkt subseq nr.gz clps_hits_id.txt > clps_hits.fasta

# This will yield the desired protein sequence in FASTA format.
```

8. Finally, by using a different approach via local BLAST, users can acquire the two necessary documents (e.g., BLASTP_clps.tsv and clps_hits.fasta) for the TreeTuner pipeline. These are files comparable to the ones generated by the NCBI server.

⚠ CRITICAL: Make sure to remove the header of the BLAST hits file (e.g., BLASTP_clps.tsv) obtained from the online BLAST. Otherwise, the annotations will confuse the scripts in the following step. Troubleshooting 2

```
# blastp

# Iteration: 0

# Query: AT1G68660.1
```

```
# RID: PVFT67SX016

# Database: nr

# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start, s. end, evalue, bit score, % positives

# 1667 hits found
```

### Minimizing the redundancy and complexity of large phylogenetic datasets via coarse tuning

Timing: ~6 h (Depending on the size of the data, computing power, and Internet speed)

The point of coarse tuning is to use TreeTrimmer (Maruyama et al., 2013) to do some preliminary trimming of the phylogeny to get a rough sense of the minimal tree diversity. Users will acquire the trimmed tree file in Newick format (e.g., "clps_aligned_trimmed.newick__clps_parameter_input.in.tt0.0.tre") in steps 9–11. Since the redundant species will be trimmed based on the taxon information, users will need to use acc2tax (https://github.com/richardmleggett/acc2tax) to first pull out the hierarchical taxonomic terms for each ClpS homolog.

9. Users can first download the package acc2tax via the GitHub link (https://github.com/richardmleggett/acc2tax) and then prepare the other necessary taxa information files downloaded from the link (https://ftp.ncbi.nih.gov/pub/taxonomy/).
   a. Four necessary taxonomic information files can be acquired from the NCBI taxonomy database which is updated weekly: nodes.dmp, names.dmp, acc2tax_nucl_all.txt, and acc2tax_prot_all.txt.
   b. nodes.dmp and names.dmp files are in the taxdump.tar.gz file after decompressing.
   c. acc2tax_nucl_all.txt and acc2tax_nucl_all.txt files will need to be created by merging files in the link (https://ftp.ncbi.nih.gov/pub/taxonomy/accession2taxid/).

```
> zcat nucl_gb.accession2taxid.gz nucl_wgs.accession2taxid.gz dead_nucl.accession2taxid.gz dead_wgs.accession2taxid.gz |sort > acc2tax_nucl_all.txt

> zcat prot.accession2taxid.gz dead_prot.accession2taxid.gz | sort > acc2tax_prot_all.txt
```

   d. The acc2tax package needs to be compiled first before running. Note that acc2tax requires the NCBI protein ID to be trimmed to remove the version (e.g., NP_564937.1 was trimmed to NP_564937). Users can follow the commands below:

```
# Compile the source code

> cc -o acc2tax acc2tax.c

# Trim the protein version

>sed 's/\(.*\)\..*/\1/g' clps_hits_id.txt > clps_hits_id_trimmed.txt

# Four taxa files need to be put in the same directory (i.e., directory name: /your/directory/Acc2tax_092021/)

>chmod +x acc2tax

>./acc2tax -i clps_hits_id_trimmed.txt -p -d /your/directory/Acc2tax_092021 -o clps_taxonomic_info.txt
```

⚠ CRITICAL: Make sure to download the latest NCBI taxonomy database which is usually updated weekly. Even so, users might still be presented with a list of unmatched protein IDs (e.g., "Couldn't find: [CAG5999297]"), because some of the species do not have complete descriptions in hierarchical taxonomic terms. Troubleshooting 3

e. The "clps_taxonomic_info.txt" file will contain the hierarchical taxonomic categories for each protein ID as the example displayed here:

```
NP_564937	Eukaryota; Viridiplantae; Streptophyta; Streptophytina; Embryophyta; Tracheophyta; Euphyllophyta; Spermatophyta; Magnoliopsida; Mesangiospermae; eudicotyledons; Gunneridae; Pentapetalae; rosids; malvids; Brassicales; Brassicaceae; Camelineae; Arabidopsis; Arabidopsis thaliana
```

10. With the hierarchical taxonomic information available, the next steps are to align the sequences, trim the alignment and build a phylogenetic tree.
    a. Since the NCBI protein ID (clps_hits_id_trimmed.txt) has been trimmed in the former step, users will need to also trim the version information in the FASTA file (clps_hits.fasta) as well to maintain consistency.

```
> awk '{print $1}' clps_hits.fasta | sed 's/^\(>.*\)\..*/\1/g' > clps_hits_trimmed.fasta

# clps_hits_trimmed.fasta

>NP_564937

METAICGRLALAPSSLFNSKSGDKHLVSKGPCVNRSILMTLSTSAALGKGGGVLDKPIIEKTTPGRESEFDLRKSKKIAP

...
```

    b. Users can then apply the multiple sequence alignment tool MAFFT v7.310 (Katoh and Standley, 2013), the alignment trimming tool BMGE v1.12 (or trimAl v1.4) (Criscuolo and Gribaldo, 2010) and FastTree v2.1 (Price et al., 2010) to build a preliminary tree.

```
##1## Users can download the version 7 of MAFFT via the link (https://mafft.cbrc.jp/alignment/software/).

> mafft --auto clps_hits_trimmed.fasta > clps_aligned.fasta

##2## Users will find the BMGE v1.12 package (BMGE-1.12.tar.gz) from ftp://ftp.pasteur.fr/pub/GenSoft/projects/BMGE/.

>bmge -i clps_aligned.fasta -t AA -m BLOSUM30 -of clps_aligned_trimmed.fasta

##Optional## Users can also use another alignment trimming tool trimAl v1.4 via the link (http://trimal.cgenomics.org)

> trimal -in clps_aligned.fasta -out clps_aligned_trimmed.fasta -htmlout output1.html -gt 1

##3## Then users can build a tree via the FastTree v2.1 (http://www.microbesonline.org/fasttree/).

>FastTree clps_aligned_trimmed.fasta >clps.aligned.trimmed.newick
```

11. With the input files available, users can test the TreeTrimmer tool with their data. The example files can be found via the GitHub link (https://github.com/zx0223winner/TreeTuner).
    a. TreeTrimmer (Maruyama et al., 2013) is written in Ruby, which requires pre-installation (Ruby v2.5.1 and BioRuby v2.0.3).

```
> sudo apt install ruby

> sudo gem install bio

# Error occurs while running the old version of TreeTrimmer.

> ''treetrimmer.rb:37: warning: IO#lines is deprecated; use each_line instead''

# Error occurs while requiring the BioRuby

> require': cannot load such file – bio (LoadError)
```

⚠ CRITICAL: We suggest users download the updated TreeTrimmer script from GitHub to ensure that the TreeTuner pipeline runs correctly (https://github.com/zx0223winner/TreeTuner/tree/main/Tutorial/TreeTuner_file_examples/Step11-14_coarse_tuning). Troubleshooting 4

b. There are three necessary input files for TreeTrimmer (Maruyama et al., 2013). Sample files can be found in the GitHub "tutorial" directory under the name "TreeTuner_file_examples". Users can first prepare the TreeTrimmer parameter input file (clps_parameter_input.in). As described in the help document, TreeTrimmer will identify and remove user-defined 'redundant' sequences via the taxa information and branch cut-off parameters. If so, the orthologous sequences from closely related organisms and 'recently' evolved lineage-specific paralogs will be greatly reduced. Users can also adjust the number of representative OTUs to be retained in the TreeTrimmer parameter input file.

```
# clps_parameter_input.in

# Specify a cut-off of support values (e.g. bootstrap values or branch length values),

# either in integer (0-100) or decimal (0.0-1.0).

# Leave it blank or use default (0.0) for trees with no branch supports (Greater than this value
will trigger for removal).

cutoff=0.0

# Which taxonomic categories should be pruned? How many OTUs should be retained?

(Tab- delimited)

Bacteria      4

Archaea      3

Eukaryota      1

# How many OTUs should be retained in each clade unless specified above?

num_retained=1
```

c. The example file can be found from the GitHub. Here, the 'query_tag' was set to 564937 for the purposes of retaining the reference *A. thaliana* ClpS protein (GenBank ID: NP_564937; TAIR ID: AT1G68660.1).

```
# clps_parameter_input.in

# Query tag (optional: default is "query_tag=QUERY")

# A string in the OTU name can be specified as a tag to avoid removal

query_tag=564937

# Delimiter for categories of the taxonomic information

taxon_delimiter=;\s
```

d. Users can obtain the other two input files (e.g., "clps_aligned_trimmed.newick" and "clps_taxonomic_info.txt") from Step 10b and 9d, respectively. However, there is an additional step needed to format the input document "clps_taxonomic_info.txt" so that it can be recognized by TreeTrimmer.

```
# Simply run the following script to format.

>sed 's/,/; /g' clps_taxonomic_info.txt|sed 's/cellular organisms; //g' > clps_taxonomic_info_clean.txt

# Here is the difference between the two files.

# clps_taxonomic_info.txt

A0A2K3CNL6 cellular organisms,Eukaryota,Viridiplantae,Chlorophyta,core chlorophytes,Chlorophyceae,CS clade,Chlamydomonadales,Chlamydomonadaceae,Chlamydomonas,Chlamydomonas reinhardtii

# clps_taxonomic_info_clean.txt

A0A2K3CNL6 Eukaryota; Viridiplantae; Chlorophyta; core chlorophytes; Chlorophyceae; CS clade; Chlamydomonadales; Chlamydomonadaceae; Chlamydomonas; Chlamydomonas reinhardtii
```

e. Once all the necessary files are prepared, users can run the command as follows:

```
> ruby treetrimmer.rb sample/clps_aligned_trimmed.newick sample/clps_parameter_input.in sample/clps_taxonomic_info_clean.txt > clps_treetrimmer.out
```

f. Two output documents will be produced (e.g., "clps_treetrimmer.out" and "clps_aligned_trimmed.newick__clps_parameter_input.in.tt0.0.tre"). The first file contains the number of OTUs to be trimmed via user-defined taxa and branch cut-off. The second file is detailed in the next step.

g. The trimmed tree file is in Newick format with a long name (e.g., "clps_aligned_trimmed.newick__clps_parameter_input.in.tt0.0.tre"). The output also displays the difference before and after running TreeTrimmer (Figures 3A and 3B). In the case of our ClpS test dataset, the number of OTUs decreased from 1157 to 44.

```
# clps_treetrimmer.out

NP_564937	Eukaryota; Viridiplantae; Streptophyta; Streptophytina; Embryophyta; Tracheophyta; Euphyllophyta; Spermatophyta; Magnoliopsida; Mesangiospermae; eudicotyledons; Gunneridae; Pentapetalae; rosids; malvids; Brassicales; Brassicaceae; Camelineae; Arabidopsis; Arabidopsis thaliana	1	86

# 1 refers to the number of OTUs was retained in each supported branch of the tree

# 86 infers the number of OTUs to be collapsed under the clade.
```

### Coloring the Newick tree after coarse tuning based on the taxonomic categories

Timing: ~30 min

Coloring the tree based on the taxonomic categories is a common way to help visualize phylogenetic diversity. Users will create the image file of a trimmed tree in steps 12–14.

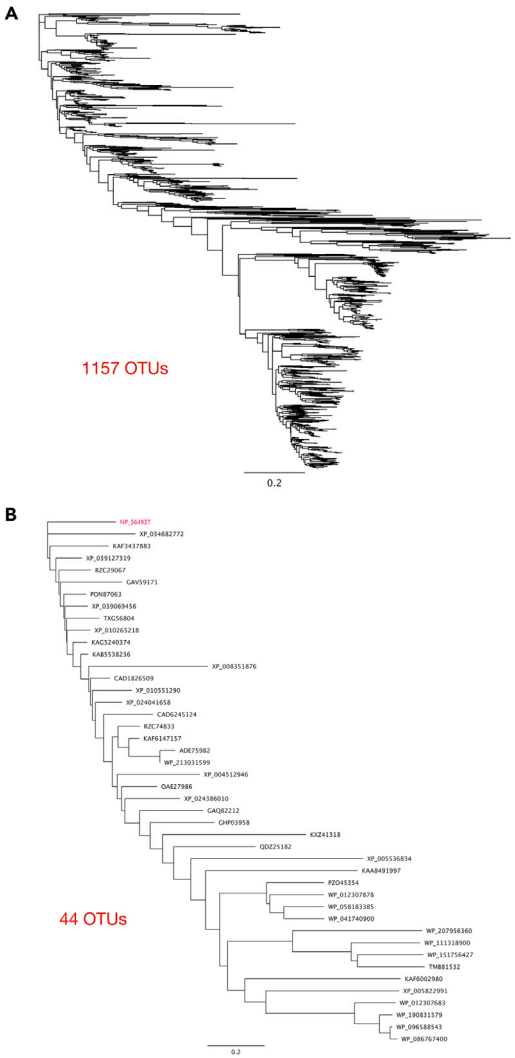



**Figure 3. The ClpS homolog phylogenetic trees before and after coarse tuning**

(A) Tree with 1157 OTUs obtained without using the coarse tuning pipeline.

(B) The 44-OTU tree retained after the use of coarse tuning pipeline. The reference gene ID: NP564937 is colored red. Phylogeny was created by FastTree (Price et al., 2010) under the maximum likelihood method model (JTT + CAT). ClpS homologs were acquired using the following settings: maximum number of BLASTP hits retrieved, 5000; BLASTP cut-off value, 1e-10.

12. Users can apply the Python script we designed to color the tree file created from step 11g.

```
# Install python3

>pip3 install python

# Install ETE3 toolkit

>pip3 install ete

# Install ETE tree browser

>pip3 install PyQt5
```

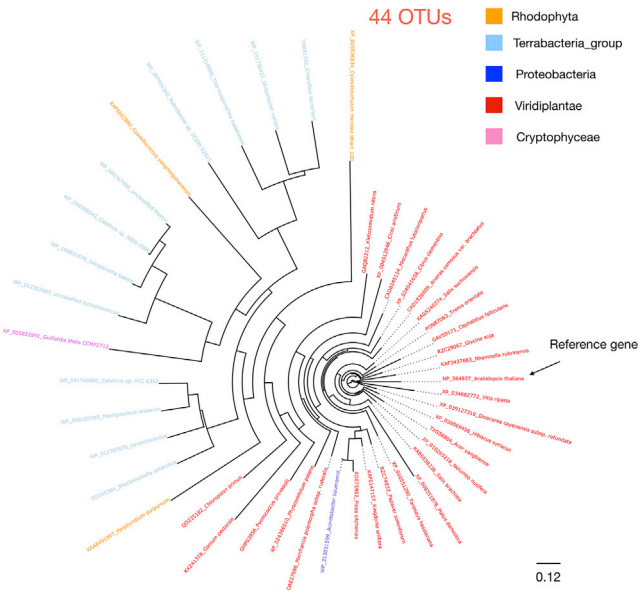



**Figure 4. ClpS protein phylogeny after coarse tuning with *Arabidopsis thaliana* homolog as query**

Phylogeny was created by FastTree (Price et al., 2010) using the 44 OTUs under the maximum likelihood method model (JTT + CAT). The coarse-tuning parameters: Cut-off value for de-replication: 0.0; Number of OTUs to be retained for each taxa categories: 4 for Bacteria, 3 for Archaea, 1 for Eukaryota. Font colors represent taxonomic categories: red for Viridiplantae; blue for Proteobacteria; yellow for Rhodophyta; light blue for Terrabacteria group; Pink for Cryptophyceae. The scale bar indicates 0.12 amino acid substitutions per site. Due to the limitations of single-gene phylogenies, the extent to which this tree reflects known or predicted organismal relationships is unclear. The phylogeny is shown solely for the purpose of illustrating the utility of the TreeTuner pipeline and should not be over-interpreted.

13. The script will require Python version 3 and ETE3 toolkit (Huerta-Cepas et al., 2016) to be pre-installed. The Python script and the relevant example files can be downloaded from the GitHub (https://github.com/zx0223winner/TreeTuner).

```
# Usage: python3 color_coarse_tuning_tree.py <taxonomic_info_file> <newick_tree_file>

# The first file is the taxonomic informatic file.

# The second file is a Newick tree file

# The colors reflect different kingdoms of the species

>python3 color_coarse_tuning_tree.py clps_taxonomic_info_clean.txt clps_aligned_
trimmed.newick__clps_parameter_input.in.tt0.0.tre

# Users can ignore the possible warnings popping up which happens when using ETE3 toolkit query
NCBI Taxonomy.

>warnings.warn("taxid %s was translated into %s" %(taxid, merged_conversion[taxid]))
```

14. Users can then visualize the tree result in the ETE tree browser after running the Python script at step 13. The different colors reflect the taxonomic categories (Figure 4).

## Minimizing the redundancy and complexity of large phylogenetic datasets via fine tuning

Timing: ~12 h

Users will have a rough sense of the taxonomic diversity contained in their phylogenetic dataset based on the results of steps 9–14. However, they may wish to proceed to some more curated,

in-depth analyses requiring more sophisticated tree-building methods and optimized taxon sampling. Here we will use the custom scripts to further resolve the complexity of the tree. Users will acquire the trimmed files in Newick format (e.g., renamed_clps_aligned_trimmed.fasta.fasttree) in steps 15–20.

15. In the same way as was done with coarse tuning, users will need to prepare two input files (e.g., "clps_hits.fasta" and "clps_acc2tax_prot_all.txt"). The first file contains the FASTA sequences for the NCBI BLAST protein homologs of the *A. thaliana* ClpS protein, which was already generated in step 7c. For the second file, users will need to do some polishing on the file acc2tax_prot_all.txt generated at step 9c.

```
# This will remove the description in the header of FASTA file.

>awk '{print $1}' clps_hits.fasta > clps_hits_no_description.fasta

# The command is to trim the large acc2tax_prot_all.txt file into smaller size with clps hits only via using the formerly generated ''clps_hits_id.txt'' file (Step 7a). There are two ways to do that.

>grep -wf clps_hits_id.txt acc2tax_prot_all.txt > clps_acc2tax_prot_all.txt

# or

>awk 'NR==FNR{a[$2]=$0;next}{print a[$1]}' acc2tax_prot_all.txt clps_hits_id.txt >clps_acc2tax_prot_all.txt
```

*Note:* Due to the size of acc2tax_prot_all.txt file (~40 Gb), it might take several hours to run this step.

16. With the two files ready, users can run the Python script called "rename_ncbi_blastdb.py" downloaded from GitHub (https://github.com/zx0223winner/TreeTuner).

```
# Usage: python3 rename_ncbi_blastdb.py <FASTA File> <Taxon Id FILE> <Renamed FASTA File>

# Note: The first time running the python script will update your local NCBI taxa database which will take around 5 mins.

# Users can comment the two lines below (Line:86 and Line:87) to avoid the updating each time.

# print("Upgrading NCBI local database...")

# ncbi.update_taxonomy_database()

>python3 rename_ncbi_blastdb.py clps_hits_no_description.fasta clps_acc2tax_prot_all.txt renamed_clps_hits.fasta

# Users can ignore the possible warnings popping up which is the same reason from the Step 9d.

#Cannot find taxid for gene HCI81531.1

#Cannot find taxid for gene MBO6986741.1

#Cannot find taxid for gene MXZ63256.1
```

*Note:* The reason for renaming the FASTA file header is to insert the hierarchical taxonomic terms. The Python script will connect the terms with an underscore ('_'), so the header is treated as a single term.

```
# Header of the FASTA file before and after renaming.

# Before

>NP_564937.1 Ribosomal protein L12/ ATP-dependent Clp protease adaptor protein ClpS family protein [Arabidopsis thaliana]

# After

# Naming Format: species_name@database_databaseID_hierarchical_taxonomic_terms

>Arabidopsis_thaliana@NCBI_NP_564937.1_Eukaryota_Viridiplantae_Streptophyta_Streptophytina_Embryophyta_Tracheophyta_Euphyllophyta_Spermatophyta_Magnoliopsida_Mesangiospermae_eudicotyledons_Gunneridae_Pentapetalae_rosids_malvids_Brassicales_Brassicaceae_Ca
```

*Note:* It is also important to remove special characters such as '/' and '()' in the header, since they can interfere with the proper running of the FastTree and Perl scripts.

```
# Create new FASTA file without special characters in the header.

#Remove ':', '(', ')', '/'

>sed 's/\///g' renamed_clps_hits.fasta |sed 's/(//g'|sed 's/)//g'|sed 's/://g' > new_renamed_clps_hits.fasta
```

17. With the renamed FASTA file, users can repeat the tree building in step 10b and change the input file name "clps_hits_trimmed.fasta" to the renamed FASTA file (e.g., "new_renamed_clps_hits.fasta").

```
##1## Users can download the version 7 of MAFFT via the link (https://mafft.cbrc.jp/alignment/software/).

> mafft –auto new_renamed_clps_hits.fasta > renamed_clps_aligned.fasta

##2## Users will find the BMGE v1.12 package (BMGE-1.12.tar.gz) from ftp://ftp.pasteur.fr/pub/GenSoft/projects/BMGE/.

>bmge -i renamed_clps_aligned.fasta -t AA -m BLOSUM30 -of renamed_clps_aligned_trimmed.fasta

##Optional## Users can also use another alignment trimming tool trimAl v1.4 via the link (http://trimal.cgenomics.org)

> trimal -in clps_aligned.fasta -out renamed_clps_aligned_trimmed.fasta -htmlout output1.html -gt 1

##3## Then users can build a tree via the FastTree v2.1 (http://www.microbesonline.org/fasttree/).

>FastTree renamed_clps_aligned_trimmed.fasta >renamed_clps_aligned_trimmed.newick

#Debugging

#Error: Non-unique name ### in the alignment

#In rare situation, users might see the errors after running the FastTree, which is because of the same header of homologs appearing in the alignment file after using the renaming script.

# Users should follow the Step 16 to remove the special characters and make sure the renamed headers are unique.
```

⚠ CRITICAL: The reason for repeating the tree building step is due to the different header formats for coarse versus fine tuning. For example, the coarse-tuning pipeline requires the header without version information (e.g., NP_564937 without '.1') to be recognized by a third-party tool (i.e., acc2tax) in step 9. Similarly, the fine-tuning pipeline requires the header containing hierarchical taxonomic terms to be pulled out by the custom scripts via taxa rank. Users should know that the tree files created by the coarse-tuning pipeline can help optimize the taxon sampling for downstream fine-tuning. Troubleshooting 5

18. The final output file from the previous step will yield a Newick tree file named "renamed_clps_aligned_trimmed.newick". We will then use two custom Perl scripts ("perl rm_inparal_rank.pl" and "trim2untrim.pl" downloaded via GitHub at https://github.com/zx0223winner/TreeTuner) to fine-tune the tree and then apply sophisticated tree-building methods and optimized taxon sampling.

```
# Usage: perl rm_inparal_rank.pl <TREE file> <Aligned_FASTA> <cut-off> <Parameter file 1> <Parameter file 3>

>perl rm_inparal_rank.pl renamed_clps_aligned_trimmed.newick renamed_clps_aligned_trimmed.fasta 1.0 taxa_not_remove.txt taxa_rank.txt

# Debugging

# Error 1: Can't locate Statistics/Descriptive.pm in @INC

# The error can be solved by running the following command:

>sudo cpan Statistics::Descriptive

# Error 2: FastTree: permission denied

# Users need to authorize the permission for FastTree and FastTree.c files. Because the Perl script 'rm_inparal_tank.pl' uses FastTree to infer the tree topology.

> gcc -DNO_SSE -O3 -finline-functions -funroll-loops -Wall - o FastTree FastTree.c -lm

> chmod +x FastTree
```

⚠ CRITICAL: While users can acquire the first two input files from the previous steps (i.e., the Newick tree file and aligned FASTA file), the remaining two parameter files will need to be carefully prepared. The output tree file created by the coarse-tuning pipeline at step 11f can be a useful guide.

```
# taxa_not_remove.txt

# This file contains a list of taxa and phyla users don't want to reduce.

# The scripts will look for matched strings listed here,

# e.g., NP_564937.1 is ID of the ClpS.

564937.1

Rhodophyta

Haptista

# taxa_rank.txt

# This file will contain information on how to reduce specific genera/phyla/kingdoms. It will reduce specifically at the taxonomic terms (0 = domain (e.g., Eukaryota), 1 = kingdom (e.g., Viridiplantae), 2 = phyla (e.g., Streptophyta), 3 = class (Klebsormidiophyceae), etc.) (space-delimited).
```

```
Bacteria      3

Archaea      2

Eukaryota      3

# Branch lengths less than this value will lead to taxon removal''.

1.0
```

19. The taxa rank file is determined based on specific header formats as follows.

```
# Eukaryotia 3

# Refers to Eukaryota_Viridiplantae_Streptophyta

>Arabidopsis_thaliana@NCBI_NP_564937.1_Eukaryota_Viridiplantae_Streptophyta_
Streptophytina_Embryophyta_Tracheophyta_Euphyllophyta_Spermatophyta_Magnoliopsida_
Mesangiospermae_eudicotyledons_Gunneridae_Pentapetalae_rosids_malvids_Brassicales_
Brassicaceae_Ca
```

20. The previous Perl script will yield three output files (e.g., "renamed_clps_aligned_trimmed.fasta_removedSeq", "renamed_clps_aligned_trimmed.fasta.fasttree", "renamed_clps_aligned_trimmed.fasta.genus_trimmed"). Then users can run the following script to acquire the trimmed sequence alignment (e.g., renamed_clps_aligned_trimmed.fasta_sub).

```
# Usage: trim2untrim.pl <Previous_output.genus_trimmed> <Aligned_FASTA>

> perl trim2untrim.pl

renamed_clps_aligned_trimmed.fasta.genus_trimmed renamed_clps_aligned_trimmed.fasta
```

21. With the fine-tuning alignment available from the previous step, users can see the difference before and after running the Perl scripts (Figures 5A and 5B). In the case of our ClpS test dataset, the number of OTUs decreased from 1144 to 66. Users can also proceed with more rigorous tree-building methods (e.g., IQ-TREE) to recreate a Newick tree file (Figure 5C).

```
# Usage: trim2untrim.pl <Previous_output.genus_trimmed>

iqtree -s renamed_clps_aligned_trimmed.fasta_sub -alrt 1000 -bb 1000 -nt AUTO
```

### Coloring the Newick tree after fine-tuning based on the taxonomic categories

Timing: ~30 min

After generating the Newick tree in the previous step, a different Python script is used to plot the tree based on the taxonomic categories, due to the fact that different headers are used by the coarse- and fine-tuning processes. Users will create the image file of a trimmed tree at steps 22–24.

22. As was the case for coloring the Newick tree after coarse-tuning, some pre-installed packages are needed.

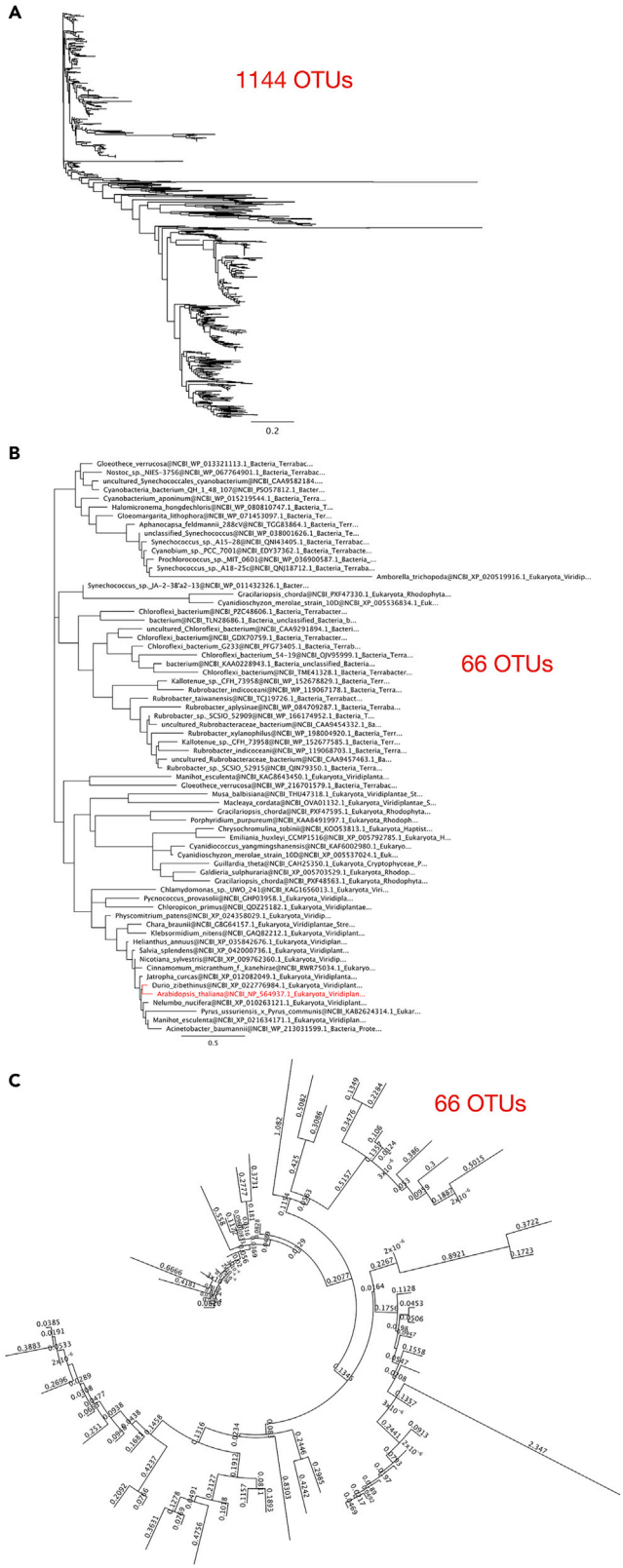

A
1144 OTUs
0.2
B
66 OTUs
0.5
C
66 OTUs

**Figure 5. ClpS protein phylogeny before and after fine-tuning**

(A) Tree containing 1144 OTUs retained before running the fine-tuning pipeline.

(B) Tree of 66 OTUs retained after using fine-tuning. The phylogeny was created by FastTree (Price et al., 2010) under the maximum likelihood method model (JTT + CAT). Fine-tuning parameters: branch length cut-off for non-removed taxa: 1.0; "taxa_not_remove.txt" listed the string name of 564937.1, Rhodophyta and Haptista; "taxa_rank.txt" file listed how to specifically reduce taxa levels at the taxonomic terms, for examples: Bacteria at 3; Archaea at 2; Eukaryota at 3.

(C) 66-OTU tree created using a more rigorous maximum-likelihood tree-building method (i.e., IQ-TREE (Nguyen et al., 2015)). The substitution model (LG + R4) used to infer the phylogenies was selected for each alignment according to the Bayesian information criterion (BIC) using ModelFinder (Kalyaanamoorthy et al., 2017). For each phylogeny, branch support was assessed using 1000 ultra-fast bootstrap approximations. The scale bar indicates the number of amino acid substitutions per site.

23. The Python script will require Python version 3 and ETE3 toolkit (Huerta-Cepas et al., 2016) to be pre-installed. The Python script and the relevant example files can be downloaded from the GitHub (https://github.com/zx0223winner/TreeTuner).

```
# Install python3

>pip3 install python

# Install ETE3 toolkit

>pip3 install ete

# Install ETE tree browser

>pip3 install PyQt5
```

24. Users can then visualize the tree in the ETE tree browser. The different colors reflect the taxa categories (Figure 6).

```
# Usage: python3 color_fine_tuning_tree.py <newick_tree_file>

# The first file is a Newick tree file

# The colors reflect different kingdoms of the species

>python3 color_fine_tuning_tree.py renamed_clps_aligned_trimmed.fasta.fasttree
```

## EXPECTED OUTCOMES

The optimal depth of taxon and sequence sampling for a molecular phylogenetic study often depends on the question being asked. Sometimes researchers require only select representatives from one particular lineage, and maximum sequence diversity from another lineage. The TreeTuner pipeline is designed for such cases. It is composed of two parts, an initial coarse-tuning step followed by fine-tuning. After the coarse-tuning pipeline is run, users will have two files, the first of which contains the sequences to be retained after OTU trimming. The second file is a trimmed tree file which can be used as input for the subsequent fine-tuning pipeline. For example, after running the coarse-tuning pipeline, users will get a rough idea of the OTUs that exist for specific lineages. Users can then apply the fine-tuning scripts to pull out sequences from lineages of particular interest to proceed with more rigorous and computer-intensive downstream analyses, such as Bayesian and maximum likelihood tree reconstructions (Maruyama et al., 2013). Users can also set different cut-offs for the branch length values. This is useful for reducing sequence redundancy from closely related organisms or the presence of lineage-specific paralogs. In order to acquire the appropriate OTU diversity for the phylogenetic question at hand, users will need to be familiar

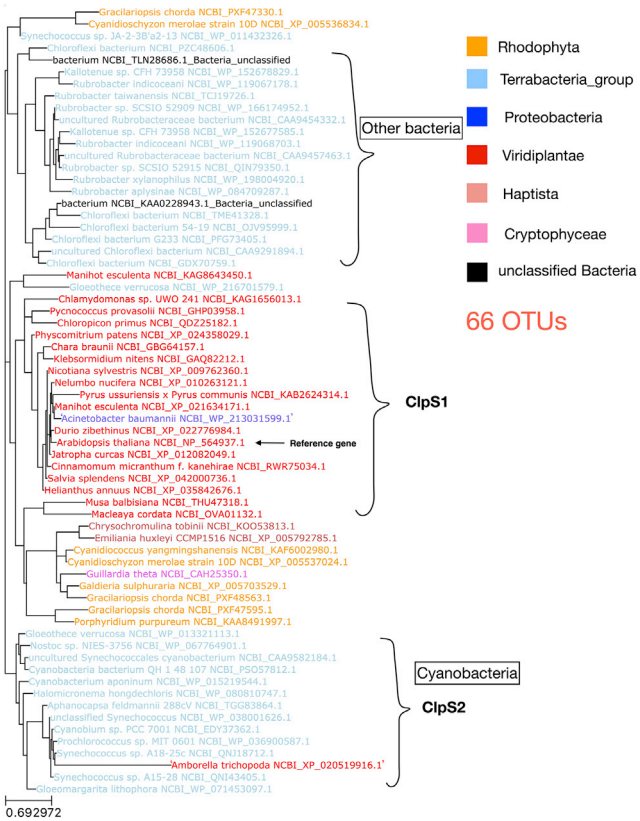



**Figure 6. ClpS protein phylogeny after TreeTuner fine tuning with *Arabidopsis thaliana* homolog as query**

The phylogeny was created by FastTree (Price et al., 2010) using the 66 OTUs under the maximum likelihood method model (JTT + CAT). The tree was visualized in the ETE tree browser (Huerta-Cepas et al., 2016). The CLPS homologs in photosynthetic organisms split into CLPS1 (all photosynthetic organisms) and CLPS2 homologs (only cyanobacteria) (Nishimura et al., 2013). The fine-tuning parameters were as follows: branch length cut-off for non-removed taxa: 1.0; "taxa_not_remove.txt" listed the string name of 564937.1, Rhodophyta and Haptista; "taxa_rank.txt" file listed how to specifically reduce taxa levels at the taxonomic terms, for examples: Bacteria at 3; Archaea at 2; Eukaryota at 3. Font colors represent taxonomic categories: yellow for Rhodophyta; light blue for Terrabacteria group; red for Viridiplantae; blue for Proteobacteria; light red for Haptista. The scale bar indicates 0.7 amino acid substitutions per site. Due to the limitations of single-gene phylogenies, the extent to which this tree reflects known or predicted organismal relationships is unclear. The phylogeny is shown solely for the purpose of illustrating the utility of the TreeTuner pipeline and should not be over-interpreted.

with their species of interest and equipped with general background knowledge of hierarchical taxonomic terms.

## LIMITATIONS

Phylogeny-based OTU reduction methods consider the phylogenetic relationship between sequences in an alignment. For example, one can set the threshold for bootstrap branch support to reflect the robustness of the tree topology and/or consider branch lengths to measure the extent of sequence divergence via measuring the average number of nucleotide or protein substitutions per site. At the same time, taxonomy-based dataset trimming can result in biased OTU retention because highly diverse clades may be represented by more leaves than less diverse ones. Also, the TreeTuner pipeline is not fully automated and relies on user-defined OTUs.

It also should be noted that there are many reasons why gene trees do not equal 'species trees', including endosymbiosis (and endosymbiotic gene transfer), lateral gene transfer, gene duplication

and differential gene loss, and incomplete lineage sorting. If we also consider contamination and gene/protein misannotation, then it can be very difficult to decide how to flag anomalies in a way that is consistent and useful. Nevertheless, we provide a step-by-step solution to guide users who need to trim down their large phylogenetic datasets for more rigorous downstream analysis. In the future, it might be possible for TreeTuner to be updated to work with other types of sequences, such as non-coding RNAs (ncRNAs), though such RNAs evolve extremely quickly and thus are not likely to retain useful phylogenetic information over large evolutionary timescales.

## TROUBLESHOOTING

### Problem 1

Why does BLASTP need to be chosen as an option? What E-value shall I choose? (step 5)

### Potential solution

Make sure to use the amino acid sequences and BLASTP option, because amino acid sequences are generally more highly conserved than their corresponding nucleotide sequences. We recommend the E-value to be no larger than 1e-10 to limit erroneous homology inferences.

### Problem 2

Why is the step of removing the specific lines necessary? (step 8)

### Potential solution

Make sure to remove the top annotated lines ('#') in BLAST hits file (e.g., BLASTP_clps.tsv) generated via the online BLAST. Otherwise, the annotations will not be recognized by the subsequent scripts.

### Problem 3

Do I have to update the NCBI taxonomy database for each run? Why do some unmatched protein IDs pop up even with the latest NCBI taxonomy database? (step 9)

### Potential solution

Make sure to download the latest NCBI taxonomy database which is usually updated weekly. But it does not have to be updated each time. Users might still get a list of unmatched protein IDs (e.g., "Couldn't find: [CAG5999297]"), this is largely due to some of the species not having the full complement of taxonomic information.

### Problem 4

Which version should I use for the TreeTrimmer in the coarse-tuning pipeline? (step 11)

### Potential solution

If users download the original ReadMe file and script of TreeTrimmer (TreeTrimmer.v130413.zip) from the link (https://code.google.com/archive/p/treetrimmer/), there might be an error after running, which might be due to the issue of version compatibility. So, we suggest users download the updated TreeTrimmer script from GitHub to use this TreeTuner pipeline. Make sure to have the pre-installation packages (e.g., Ruby v2.5.1 and BioRuby v2.0.3) before running the TreeTrimmer.

### Problem 5

Why do I have to rebuild the phylogenetic tree? Can I still use the tree file from the coarse-tuning pipeline? (step 17)

### Potential solution

The reason for repeating the tree building step is due to the different header format between coarse and fine tuning. For example, the coarse-tuning pipeline requires the header without version (e.g., NP_564937 without '.1') to be recognized by the third-party tool (i.e., acc2tax) at step 9. Similarly,

the fine-tuning pipeline requires the header containing hierarchical taxonomic terms to be pulled out by the custom scripts via taxa rank. Users should know that tree files created by coarse-tuning pipeline can help the fine-tuning to optimize the taxon sampling.

## RESOURCE AVAILABILITY

### Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact John M. Archibald (john.archibald@dal.ca) and Technical Contact Xi Zhang (xi.zhang@dal.ca).

### Materials availability

This study did not generate new unique reagents.

### Data and code availability

The TreeTuner source code has been deposited at https://github.com/zx0223winner/TreeTuner.

## ACKNOWLEDGMENTS

This work was supported by a Gordon and Betty Moore Foundation grant (GBMF5782) and by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada (RGPIN-2019-05058), both awarded to J.M.A. We thank all authors of the papers that were cited in this manuscript. We also thank the editors and reviewers for their helpful comments on an earlier version of this manuscript.

## AUTHOR CONTRIBUTIONS

The study was conceptualized by X.Z., K.J.V.W., and J.M.A. The data and manuscript were analyzed and written by X.Z. Y.N.H. assisted with bioinformatics analysis and debugged the TreeTuner pipeline. L.E. wrote the Perl script which was used in the fine-tuning pipeline of TreeTuner. S.M. and R.J.M.E. developed the TreeTrimmer algorithm which was used in the coarse-tuning pipeline of TreeTuner. S.J.S. and B.A.C. provided valuable suggestions and discussions. All authors read, revised, and approved the final manuscript.

## DECLARATION OF INTERESTS

The authors declare no competing interests.

## REFERENCES

Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. *25*, 3389–3402.

Bouchnak, I., and van Wijk, K.J. (2019). N-degron pathways in plastids. Trends Plant Science *24*, 917–926.

Capella-Gutiérrez, S., Silla-Martínez, J.M., and Gabaldón, T. (2009). TrimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. Bioinformatics *25*, 1972–1973.

Caron, D.A., Alexander, H., Allen, A.E., Archibald, J.M., Armbrust, E.V., Bachy, C., Bell, C.J., Bharti, A., Dyhrman, S.T., and Guida, S.M. (2017). Probing the evolution, ecology and physiology of marine protists using transcriptomics. Nat. Rev. Microbiol. *15*, 6–20.

Cheng, C.Y., Krishnakumar, V., Chan, A.P., Thibaud-Nissen, F., Schobel, S., and Town, C.D. (2017). Araport11: a complete reannotation of the *Arabidopsis thaliana* reference genome. Plant J. *89*, 789–804.

Criscuolo, A., and Gribaldo, S. (2010). BMGE (Block Mapping and Gathering with Entropy): a new software for selection of phylogenetic informative regions from multiple sequence alignments. BMC Evol. Biol. *10*, 1–21.

Emms, D., and Kelly, S. (2021). SHOOT: phylogenetic gene search and ortholog inference. bioRxiv. https://doi.org/10.1101/2021.09.01.458564.

Federhen, S. (2012). The NCBI taxonomy database. Nucleic Acids Res. *40*, D136–D143.

Huerta-Cepas, J., Serra, F., and Bork, P. (2016). ETE 3: reconstruction, analysis, and visualization of phylogenomic data. Mol. Biol. Evol. *33*, 1635–1638.

Kalyaanamoorthy, S., Minh, B.Q., Wong, T.K., Von Haeseler, A., and Jermiin, L.S. (2017). ModelFinder: fast model selection for accurate phylogenetic estimates. Nat. Methods *14*, 587–589.

Katoh, K., and Standley, D.M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol. Biol. Evol. *30*, 772–780.

Keeling, P.J., Burki, F., Wilcox, H.M., Allam, B., Allen, E.E., Amaral-Zettler, L.A., Armbrust, E.V., Archibald, J.M., Bharti, A.K., and Bell, C.J. (2014). The Marine Microbial Eukaryote Transcriptome Sequencing Project (MMETSP): illuminating the functional diversity of eukaryotic life in the oceans through transcriptome sequencing. PLoS Biol. *12*, e1001889.

Krishnamoorthy, M., Patel, P., Dimitrijevic, M., Dietrich, J., Green, M., and Macken, C. (2011). Tree pruner: an efficient tool for selecting data from a biased genetic database. BMC Bioinformatics *12*, 1–8.

Maruyama, S., Eveleigh, R.J., and Archibald, J.M. (2013). Treetrimmer: a method for phylogenetic dataset size reduction. BMC Res. Notes *6*, 1–6.

Menardo, F., Loiseau, C., Brites, D., Coscolla, M., Gygli, S.M., Rutaihwa, L.K., Trauner, A., Beisel, C., Borrell, S., and Gagneux, S. (2018). Treemmer: a tool to reduce large phylogenetic datasets with minimal loss of diversity. BMC Bioinformatics *19*, 1–8.

Nguyen, L.-T., Schmidt, H.A., Von Haeseler, A., and Minh, B.Q. (2015). IQ-TREE: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. Mol. Biol. Evol. *32*, 268–274.

Nishimura, K., Asakura, Y., Friso, G., Kim, J., Oh, S.-h., Rutschow, H., Ponnala, L., and van Wijk, K.J. (2013). ClpS1 is a conserved substrate selector for the chloroplast Clp protease system in Arabidopsis. Plant Cell *25*, 2276–2301.

Price, M.N., Dehal, P.S., and Arkin, A.P. (2010). FastTree 2–approximately maximum-likelihood trees for large alignments. PLoS One *5*, e9490.

Pruitt, K.D., Tatusova, T., and Maglott, D.R. (2005). NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. Nucleic Acids Res. *33*, D501–D504.

Schoch, C.L., Ciufo, S., Domrachev, M., Hotton, C.L., Kannan, S., Khovanskaya, R., Leipe, D., Mcveigh, R., O'Neill, K., and Robbertse, B. (2020). NCBI Taxonomy: a comprehensive update on curation, resources and tools. Database. https://doi.org/10.1093/database/baaa062.

Shen, W., Le, S., Li, Y., and Hu, F. (2016). SeqKit: a cross-platform and ultrafast toolkit for FASTA/Q file manipulation. PLoS One *11*, e0163962.

Sibbald, S.J., Hopkins, J.F., Filloramo, G.V., and Archibald, J.M. (2019). Ubiquitin fusion proteins in algae: implications for cell biology and the spread of photosynthesis. BMC Genomics *20*, 1–13.

Yandell, M., and Ence, D. (2012). A beginner's guide to eukaryotic genome annotation. Nat. Rev. Genet. *13*, 329–342.

Zhang, X., Hu, Y., and Smith, D.R. (2021a). Protocol for HSDFinder: identifying, annotating, categorizing, and visualizing duplicated genes in eukaryotic genomes. STAR Protoc. *2*, 100619.

Zhang, X., Hu, Y., and Smith, D.R. (2021b). Protocol for using NoBadWordsCombiner to merge and minimize "bad words" from BLAST hits against multiple eukaryotic gene annotation databases. STAR Protoc. *2*, 100888.