Phylogenomic analysis of Thalassiosirales genomes and transcriptomes

Home directory on Pinnacle:

`/storage/wader/Genome-Skimming-annotation/`

Software used:

- OrthoFinder (v.2.4.0)
- Mafft (v.7.304b)
- TrimAl (v.1.4rev22)
- IQ-TREE (v.1.6.12)
- IQ-TREE2 (v.2.0.3)
- ASTRAL (v.5.7.3)
- FastTree (v.2.1.10)
- TreeShrink (v.1.3.7)
- Pal2Nal (v.14)
- PhyKit (v.1.2.1)
- PAML (v.4.9e)
- QuartetSampling (v.1.3.1)
- Yang & Smith pipeline scripts (`https://bitbucket.org/yanglab/phylogenomic_dataset_construction/src/master/`)
- GeneWise (`https://www.ebi.ac.uk/Tools/psa/genewise/`)
- AMAS (`https://github.com/marekborowiec/AMAS`)
- Degen (`http://www.phylotools.com/ptdegendownload.htm`)
- MSA_trimmer (`https://github.com/LKremer/MSA_trimmer`)
- extract-codon-alignment (`https://github.com/linzhi2013/extract_codon_alignment`)

---

1. Run OrthoFinder to get orthogroups and create sequence files

```
orthofinder -t 24 -a 12 -og -f orthofinder-proteomes
```

Summarize the orthogroup occupancy

Working directory:

`orthofinder-proteomes/OrthoFinder/Results_Mar29/Orthogroups/`

```
python3 summarize_orthogroup_membership.py Orthogroups.GeneCount.tsv outgroup-species
.txt Coscinodiscus_sp_AJA212-04 > orthogroup-summary.csv
```

Select and link to orthogroups with >= 20% ingroup occupancy

Working directory:

`OrthoFinder/orthogroup_fasta_occupancy_40/`

```
python3 link_to_orthogroups_by_taxon_occupancy.py {path_to}/orthogroup-summary.csv {p
ath_to}/Orthogroup_Sequences/
```

Selected 11687 orthogroups with >= 20% ingroup occupancy

---

2. Multiple sequence alignment, Round 1

Working directory:

`OrthoFinder/orthogroup_fasta_occupancy_40/`

Orthogroups with <= 200 sequences will be aligned using Mafft

```
mafft --anysymbol --genafpair --maxiterate 1000 --thread 4 INPUT > OUTPUT
```

For orthogroups with > 200 sequences, use --auto option in Mafft

Create a list of files and split into chunks of 400

```
for f in `cat orthogroups-below200.txt` ; do python3 write_mafft_PBS_scripts.py $f -w
 4 -q tiny16core > ${f%.fa}.mafft.pbs ; done

ls *.mafft.pbs > orthogroups-below200-list.txt

split -l 400 -d orthogroups-below200-list.txt orthogroups-below200-list
```

Submit PBS jobs for each chunk

```
for f in `cat orthogroups-below200-list00` ; do qsub ${f} ; done
```

Progress of above200 chunks:

Chunk	Node	Progress
00	tiny16core	done
01	tiny16core	done

Progress of below200 chunks:

Chunk	Node	Progress
00	tiny12core	done
01	tiny12core	done
02	tiny12core	done
03	tiny12core	done
04	tiny12core	done
05	tiny12core	done
06	tiny12core	done
07	tiny12core	done
08	tiny12core	done
09	tiny12core	done
10	tiny12core	done
11	tiny12core	done
12	tiny12core	done
13	tiny12core	done
14	tiny12core	done

15	tiny16core	done
16	tiny16core	done
17	tiny16core	done
18	tiny12core	done
19	tiny12core	done
20	tiny12core	done
21	tiny12core	done
22	tiny12core	done
23	tiny16core	done
24	tiny12core	done
25	tiny12core	done
26	tiny16core	done
27	tiny12core	done

Trim the alignments using TrimAl

```
trimal -in INPUT -out OUTPUT -gt 0.1
```

Output directory:

`OrthoFinder/orthogroups_alignments_occupancy_40/`

11687 homolog alignments

---

3. Build homolog trees, Round 1

Working directory:

`OrthoFinder/orthogroup_alignments_occupancy_40/`

For orthogroups < 1000 sequences, estimate trees using IQ-Tree

```
iqtree -s INFILE -pre PREFIX -st AA -nt AUTO -m TEST --runs 1 > INFILE.err
```

For orthogroups >= 1000 sequences, estimate trees using FastTree

```
FastTree -wag -gamma INPUT > OUTPUT
```

Output directory:

`OrthoFinder/orthogroup_trees_occupancy_40/`

Create a list of files and split into chunks of 100

```
for f in `cat alignment-below1000.txt` ; do python3 write_IQ-TREE_AA_tree_PBS_scripts
.py $f tiny12core > ${f%.aln.trim}.iqtree.pbs ; done

ls *.fasttree.pbs > alignment-below1000-submit.txt

split -l 400 -d alignment-below1000-submit.txt alignment-below1000-submit
```

Submit PBS jobs for each chunk

```
for f in `cat alignment-below1000-submit00` ; do qsub ${f} ; done
```

Progress of above1000 chunks:

Chunk	Node	Progress
00	tiny16core	done

Progress of below1000 chunks:

Chunk	Node	Progress
00	tiny12core	done
01	tiny16core	done
02	tiny12core	done
03	tiny16core	done

04	tiny12core	done
05	tiny12core	done
06	tiny16core	done
07	tiny12core	done
08	tiny12core	done
09	tiny16core	done
10	tiny12core	done
11	tiny12core	done
12	tiny16core	done
13	tiny12core	done
14	tiny16core	done
15	tiny12core	done
16	tiny12core	done
17	tiny16core	done
18	tiny12core	done
19	tiny16core	done
20	tiny12core	done
21	tiny12core	done
22	tiny12core	done
23	tiny12core	done
24	tiny12core	done
25	tiny12core	done
26	tiny12core	done
27	tiny12core	done
28	tiny12core	done

29	tiny12core	done

Check for incomplete tree searches and continue them:

```
grep -L "Analysis results written to" OG*.err > continue-these.txt
```

Final: 11687 homolog trees

---

4. Trim homolog trees, Round 1

Working directory:

`/OrthoFinder/orthogroup_trees_occupancy_40/`

Trim spurious tips using TreeShrink (quantile = 0.01)

```
conda activate treeshrink
module load R
python2 tree_shrink_wrapper.py inDIR .tree 0.01 outDIR
```

Output are `.tt` files

Summarize the TreeShrink output to check for overtrimming of outgroup sequences

```
python3 summarize_treeshrink.py .tree.txt .
```

Outgroup tips removed by quantile

taxon	start	0.05	0.01	0.005	0.001
Coscinodiscus	15051	888	241	140	51
Bellerochea	13227	205	42	18	14
Lithodesmium	13940	253	63	40	20
Ditylum	12945	236	53	32	13
Eunotogramma	18321	474	139	95	46

Mask monophyletic and paraphyletic tips

```
python2 mask_tips_by_taxonID_transcripts.py inDIR alnDIR y
```

Output are `.mm` files.

Select 10 random tree files to determine appropriate internal branch length cutoff

```
ls *.mm | shuf -n 10
```

Cut deep paralogs

```
python2 cut_long_internal_branches.py inDIR .mm 0.5 20 outDIR
```

Output directory:

`/OrthoFinder/orthogroup_subtrees_occupancy_40/`

Output are `.subtree` files

Output is 11694 subtrees

---

5. Multiple sequence alignment, Round 2

Write new fasta files for each subtree

Working directory:

`/OrthoFinder/orthogroup_subtrees_occupancy_40/`

Output directory:

`/OrthoFinder/orthogroup_fasta_round2/`

```
python2 write_fasta_files_from_trees.py all-proteomes.faa inDIR .subtree outDIR
```

Orthogroups with <= 200 sequences will be aligned using Mafft

```
mafft --anysymbol --genafpair --maxiterate 1000 --thread 4 INPUT > OUTPUT
```

For orthogroups with > 200 sequences, use --auto option in Mafft

Create a list of files and split into chunks of 100

```
for f in `cat homolog-below200.txt` ; do python3 write_mafft_PBS_scripts.py $f -w 4 -
q tiny16core > ${f%.fa}.mafft.pbs ; done

ls *.mafft.pbs > homolog-below200.txt

split -l 400 -d homolog-below200.txt homolog-below200-
```

Submit PBS jobs for each chunk

```
for f in `cat homolog-below200-00` ; do qsub ${f} ; done
```

Progress of above200 chunks:

Chunk	Node	Progress
00	tiny16core	done

Progress of below200 chunks:

Chunk	Node	Progress
00	tiny16core	done
01	tiny16core	done
02	tiny16core	done
03	tiny16core	done
04	tiny16core	done
05	tiny16core	done
06	tiny16core	done
07	tiny16core	done
08	tiny16core	done
09	tiny16core	done

10	tiny16core	done
11	tiny16core	done
12	tiny16core	done
13	tiny16core	done
14	tiny12core	done
15	tiny12core	done
16	tiny12core	done
17	tiny12core	done
18	tiny12core	done
19	tiny12core	done
20	tiny12core	done
21	tiny12core	done
22	tiny12core	done
23	tiny12core	done
24	tiny12core	done
25	tiny12core	done
26	tiny12core	done
27	tiny12core	done
28	tiny12core	done

Output is 11694 alignments

---

6. Build homolog trees, Round 2

Working directory:

`OrthoFinder/orthogroup_alignments_round2/`

For orthogroups < 1000 sequences, estimate trees using IQ-Tree

```
iqtree -s INFILE -pre PREFIX -st AA -nt AUTO -m TEST --runs 1 > INFILE.err
```

For orthogroups >= 1000 sequences, estimate trees using FastTree

```
FastTree -wag -gamma INPUT > OUTPUT
```

Output directory:

`OrthoFinder/orthogroup_trees_round2/`

Create a list of files and split into chunks of 100

```
for f in `cat alignment-below1000.txt` ; do python3 write_IQ-TREE_AA_tree_PBS_scripts
.py $f tiny12core > ${f%.aln.trim}.iqtree.pbs ; done

ls *.fasttree.pbs > alignment-below1000-submit.txt

split -l 400 -d alignment-below1000-submit.txt alignment-below1000-submit
```

Submit PBS jobs for each chunk

```
for f in `cat alignment-below1000-submit00` ; do qsub ${f} ; done
```

Progress of above1000 chunks:

Chunk	Node	Progress
00	tiny12core	done

Progress of below1000 chunks:

Chunk	Node	Progress
00	tiny12core	done
01	tiny12core	done
02	tiny12core	done

03	tiny12core	done
04	tiny12core	done
05	tiny12core	done
06	tiny12core	done
07	tiny12core	done
08	tiny12core	done
09	tiny12core	done
10	tiny12core	done
11	tiny12core	done
12	tiny16core	done
13	tiny12core	done
14	tiny12core	done
15	tiny16core	done
16	tiny12core	done
17	tiny16core	done
18	tiny12core	done
19	tiny12core	done
20	tiny16core	done
21	tiny12core	done
22	tiny12core	done
23	tiny12core	done
24	tiny12core	done
25	tiny12core	done
26	tiny12core	done
27	tiny12core	done

28	tiny12core	done
29	tiny12core	done

Check for incomplete tree searches and continue them:

```
grep -L "Analysis results written to" OG*.log > continue-these.txt
```

IQTree did not finish for 3 orthogroups. Run these using FastTree: OG0000042_1rr (JTT), OG0000080_1rr (WAG), OG0000186_1rr (WAG)

Final: 11694 homolog trees

---

7. Trim homolog trees, Round 2

Working directory:

`/OrthoFinder/orthogroup_trees_round2/`

Trim spurious tips using TreeShrink (quantile = 0.01)

```
conda activate treeshrink
module load R
python2 tree_shrink_wrapper.py inDIR .tree 0.01 outDIR
```

Output are `.tt` files

Summarize the TreeShrink output to check for overtrimming of outgroup sequences

```
python3 summarize_treeshrink.py .tree.txt .
```

Outgroup tips removed by quantile

taxon	start	0.01	0.005	0.001
Coscinodiscus	7048	325	194	62
Bellerochea	5340	34	25	14
Lithodesmium	5429	35	21	9
Ditylum	5400	43	23	12
Eunotogramma	8386	162	93	33

Mask monophyletic and paraphyletic tips

```
python2 mask_tips_by_taxonID_transcripts.py inDIR alnDIR y
```

Output are `.mm` files.

Select 10 random tree files to determine appropriate internal branch length cutoff

```
ls *.mm | shuf -n 10
```

Cut deep paralogs

```
python2 cut_long_internal_branches.py inDIR .mm 0.5 20 outDIR
```

Output directory:

`/OrthoFinder/orthogroup_subtrees_round2/`

Output are `.subtree` files

Output is 11635 subtrees

Summarize homolog occupancy

```
python2 ortholog_occupancy_stats.py .
```

---

8. Prune orthologs

Working direcctory:

`/OrthoFinder/orthogroup_subtrees_round2/`

Rooted Ingroup (RT) method: Coscinodiscus is outgroup.

```
python2 prune_paralogs_RT.py inDIR .subtree outDIR 22 taxonIDs.txt
```

Output directory:

`/OrthoFinder/orthogroup_RT_orthologs/`

Output are `.tre` files

Output is 6262 orthologs

Summarize ortholog occupancy

```
python2 ortholog_occupancy_stats.py .
```

Write new fasta files for each ortholog

```
python2 write_ortholog_fasta_files.py all-proteomes.faa orthoDIR outDIR 22
python2 write_ortholog_fasta_files.py allCDS.fna orthoDIR outDIR 22
```

Output directories:

`/OrthoFinder/orthogroup_RT_AA_fasta/` `/OrthoFinder/orthogroup_RT_CDS_fasta/`

Stupid NA51C2 has duplicates in fasta headers

`awk 'match($0,/WR35_[0-9]+-R[0-9]/,a){print $0"_"a[0]}' test.cds > test.cds2`
`sed -i.bak 's|\(_WR35_[0-9]+-R[0-9]\)|&\1|' test.cds`
`cut -d '_' --fields=1,2,3,4,5 test.faa > test.faa2`

9. Final multiple sequence alignment: orthologs

Remove `Thalassiosira_weissflogii_AJA159-02` and from fasta files

```
python3 filter_fasta_from_list.py INFILE exclude-list > OUTFILE
```

AA sequences

Working directory:

`/OrthoFinder/orthogroup_RT_AA_fasta/`

```
mafft --anysymbol --localpair --maxiterate 1000 INPUT > OUTPUT
```

Output directory:

`/OrthoFinder/orthogroup_RT_AA_alignments/`

```
for f in *.cut.faa ; do python3 write_mafft_PBS_scripts.py $f -w 4 -q tiny12core > ${
f%.cut.faa}.mafft.pbs ; done

ls *.mafft.pbs > fasta-list.txt

split -l 400 -d fasta-list.txt fasta-list

for f in `cat fasta-list00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
00	tiny12core	done
01	tiny12core	done
02	tiny12core	done
03	tiny12core	done
04	tiny12core	done
05	tiny12core	done
06	tiny12core	done
07	tiny12core	done
08	tiny12core	done
09	tiny12core	done
10	tiny12core	done
11	tiny12core	done
12	tiny12core	done
13	tiny12core	done
14	tiny12core	done
15	tiny12core	done

Trim the AA alignments

```
for i in *.aln ; do python3 alignment_trimmer.py -p $i --trim_gappy 0.9 ; done

for i in *_trimmed.aln ; do mv $i ${i%_trimmed.aln}.aln.trim ; done
```

Total trimmed AA alignments: 6262

CDS sequences

Working directory:

`/OrthoFinder/orthogroup_RT_CDS_fasta/`

```
pal2nal pep.aln CDS.fasta -output fasta > OUTPUT
```

Output directory:

`/OrthoFinder/orthogroup_RT_CDS_alignments/`

```
for f in *.cut.cds ; do python3 write_pal2nal_PBS_scripts.py $f -w 4 -q tiny12core >
${f%.cut.cds}.pal2nal.pbs ; done

ls *.pal2nal.pbs > fasta-list.txt

split -l 400 -d fasta-list.txt fasta-list

for f in `cat fasta-list00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
00	tiny12core	done
01	tiny12core	done
02	tiny12core	done
03	tiny12core	done
04	tiny12core	done
05	tiny12core	done
06	tiny12core	done
07	tiny12core	done
08	tiny12core	done
09	tiny12core	done
10	tiny12core	done
11	tiny12core	done
12	tiny12core	done
13	tiny12core	done
14	tiny12core	done
15	tiny12core	done

Inconsistency errors from `CCMP1335` in the following orthologs. Check with GeneWise, fix erros, and re-run Pal2Nal.

ortholog	fixed
OG0004774*1rr*1.inclade1.ortho1	Y
OG0002536*1rr*1.inclade1.ortho1	Y
OG0001037*1rr*1.inclade1.ortho1	Y
OG0003914*1rr*1.inclade1.ortho1	Y

OG0000272*1rr*1.inclade1.ortho1	Y
OG0004376*1rr*1.inclade1.ortho1	Y
OG0000616*2rr*1.inclade1.ortho2	Y
OG0001926*1rr*1.inclade1.ortho1	Y
OG0000229*1rr*1.inclade1.ortho2	Y
OG0003300*1rr*1.inclade1.ortho1	Y
OG0003628*1rr*1.inclade1.ortho1	Y
OG0006547*1rr*1.inclade1.ortho1	Y
OG0010405*1rr*1.inclade1.ortho1	Y
OG0001250*2rr*1.inclade1.ortho1	Y
OG0000354*4rr*1.inclade1.ortho1	Y
OG0001530*1rr*1.inclade1.ortho1	Y
OG0004850*1rr*1.inclade1.ortho1	Y
OG0004655*1rr*1.inclade1.ortho1	Y
OG0009591*1rr*1.inclade1.ortho1	Y
OG0004936*1rr*1.inclade1.ortho1	Y
OG0007207*1rr*1.inclade1.ortho1	Y
OG0009393*1rr*1.inclade1.ortho1	Y
OG0002637*1rr*1.inclade1.ortho1	Y
OG0001471*1rr*1.inclade1.ortho1	Y
OG0004069*1rr*1.inclade1.ortho2	Y
OG0001170*1rr*1.inclade1.ortho2	Y
OG0001200*1rr*1.inclade1.ortho1	Y
OG0003872*1rr*1.inclade1.ortho1	Y
OG0000086*1rr*1.inclade1.ortho2	Y

Trim the CDS alignments

```
for i in *.pal2nal ; do python3 alignment_trimmer.py -c $i --trim_gappy 0.9 ; done

for i in *_trimmed.pal2nal ; do mv $i ${i%_trimmed.pal2nal}.pal2nal.trim ; done
```

Total trimmed CDS alignments: 6262

---

10. Final tree estimation: orthologs

AA alignments

Working directory:

`/OrthoFinder/orthogroup_RT_AA_alignments/`

```
iqtree -s INPUT -pre PREFIX -st AA -nt AUTO -m TEST --runs 5 -bb 1000 -wbt > PREFIX.e
rr
```

```
for i in *.aln.trim ; do python3 write_IQ-TREE_AA_tree_PBS_scripts.py $i tiny12core >
 ${i%.aln.trim}.iqtree.pbs ; done

ls *.iqtree.pbs > alignment-list.txt

split -l 400 -d alignment-list.txt alignment-list

for f in `cat alignment-list00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
00	tiny12core	done
01	tiny12core	done
02	tiny12core	done
03	tiny12core	done
04	tiny12core	done
05	tiny12core	done
06	tiny16core	done
07	tiny12core	done
08	tiny12core	done
09	tiny16core	done
10	q06h32c	done
11	tiny16core	done
12	tiny12core	done
13	q06h32c	done
14	q06h32c	done
15	tiny16core	done

Check for incomplete tree searches and continue them:

```
grep -L "Analysis results written to" OG00*.log
```

CDS alignments - partition by codon position

Working directory:

`/OrthoFinder/orthogroup_RT_CDS_alignments/`

```
iqtree -s INPUT -pre PREFIX -spp PARTITION -st DNA -nt AUTO -m TESTMERGE --runs 5 -bb
 1000 -wbt > PREFIX.err
```

```
for i in *.pal2nal.trim ; do python3 write_IQ-TREE_CDS_tree_PBS_scripts.py $i -w 6 -q
 q06h32c ; done

ls *.iqtree.pbs > alignment-list.txt

split -l 400 -d alignment-list.txt alignment-list

for f in `cat alignment-list00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
00	q06h32c	done
01	q06h32c	done
02	q06h32c	done
03	q06h32c	done
04	q06h32c	done
05	q06h32c	done
06	q06h32c	done
07	q06h32c	done
08	q06h32c	done
09	q06h32c	done
10	q06h32c	done
11	q06h32c	done
12	q06h32c	done
13	q06h32c	done
14	q06h32c	done
15	q06h32c	done

Check for incomplete tree searches and continue them:

```
grep -L "Analysis results written to" OG00*.log
```

CDS alignments - first and second codon positions

Working directory:

`/OrthoFinder/orthogroup_RT_CDS_alignments/`

```
for i in *.pal2nal.trim ; do extract_codon_alignment --alignedCDS $i --codonPoses 12
--outAln outDir/$${%i.pal2nal.trim}.pos12.pal2nal.trim ; done
```

Output directory:

`/OrthoFinder/orthogroup_RT_CDS_pos12_alignments/`

```
iqtree -s INPUT -pre PREFIX -spp PARTITION -st DNA -nt AUTO -m TESTMERGE --runs 5 -bb
 1000 -wbt > PREFIX.err
```

```
for i in *.pos12.pal2nal.trim ; do python3 write_IQ-TREE_CDS_tree_PBS_scripts.py $i -
w 6 -q q06h32c ; done

ls *.iqtree.pbs > alignment-list.txt

split -l 400 -d alignment-list.txt alignment-list

for f in `cat alignment-list00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
00	q06h32c	done
01	q06h32c	done
02	q06h32c	done
03	q06h32c	done
04	q06h32c	done
05	q06h32c	done
06	q06h32c	done
07	q06h32c	done
08	q06h32c	done
09	q06h32c	done
10	tiny16core	done
11	q06h32c	done
12	tiny16core	done
13	q06h32c	done
14	q06h32c	done
15	tiny16core	done

Check for incomplete tree searches and continue them:

```
grep -L "Analysis results written to" OG00*.log
```

Codon degenerated CDS alignments

Working directory:

`/OrthoFinder/orthogroup_RT_CDS_alignments_DEGEN`

Create degenerated alignments using DEGEN

```
for i in ../orthogroup_RT_CDS_alignments/*.pal2nal.trim ; do ln -s $i ; done

for i in *.pal2nal.trim ; do perl Degen_v1_4.pl $i ; done
```

```
iqtree -s INPUT -pre PREFIX -spp PARTITION -st DNA -nt AUTO -m TESTMERGE --runs 5 -bb
 1000 -wbt > PREFIX.err
```

```
for i in *.fasta ; do python3 write_IQ-TREE_CDS_tree_slurm_scripts.py $i -c 6 -nt 6 -
q comp06 ; done

ls *.slurm > alignment-list.txt

split -l 400 -d alignment-list.txt alignment-list

for f in `cat alignment-list00` ; do sbatch ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
00	comp06	done
01	comp06	done
02	comp06	done
03	comp06	done
04	comp06	done
05	comp06	done
06	comp06	done
07	comp06	done
08	comp06	done
09	comp06	done
10	comp06	done
11	comp06	done
12	comp06	done
13	comp06	done
14	comp06	done
15	comp06	done

---

11. Final multiple sequence alignment: homologs

Working directory:

`/OrthoFinder/orthogroup_subtrees_round2/`

Write new fasta files for each homolog

```
python2 write_fasta_files_from_trees.py all-proteomes.faa inDIR .subtree outDIR
```

Output directory:

`/OrthoFinder/orthogroup_Homologs/`

Remove `Thalassiosira_weissflogii_AJA159-02` and from fasta files

```
for i in *.fa ; do grep "Thalassiosira_weissflogii_AJA159-02" $i > ${i%.fa}.remove ;
sed -i 's/>//g' ${i%.fa}.remove ; python3 filter_fasta_from_list.py $i ${i%.fa}.remov
e > ${i%.fa}.cut.fa ; done
```

AA sequences

Working directory:

`/OrthoFinder/orthogroup_Homologs/`

For homologs that contain <= 200 sequences:

```
mafft --anysymbol --genafpair --maxiterate 1000 INPUT > OUTPUT
```

For homologs that contain > 200 sequences, use mafft `--auto` option

Output directory:

`/OrthoFinder/orthogroup_Homologs_alignments/`

```
for f in `cat fasta-below200.txt` ; do python3 write_mafft_PBS_scripts.py $f -w 4 -q
tiny12core > ${f%.cut.fa}.mafft.pbs ; done

sed -i 's/cut.fa/mafft.pbs/g' fasta-below200.txt

split -l 400 -d fasta-below200.txt fasta-below200-

for f in `cat fasta-below200-00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
above200	tiny12core	done
00	tiny12core	done

01	tiny12core	done
02	tiny12core	done
03	tiny12core	done
04	tiny12core	done
05	tiny12core	done
06	tiny12core	done
07	tiny12core	done
08	tiny12core	done
09	tiny12core	done
10	tiny12core	done
11	tiny12core	done
12	tiny12core	done
13	tiny12core	done
14	tiny12core	done
15	tiny12core	done
16	tiny12core	done
17	tiny12core	done
18	tiny12core	done
19	tiny12core	done
20	tiny12core	done
21	tiny12core	done
22	tiny12core	done
23	tiny12core	done
24	tiny12core	done

25	tiny12core	done
26	tiny12core	done
27	tiny12core	done
28	tiny12core	done

Trim the AA alignments

```
for i in *.aln ; do python3 alignment_trimmer.py -p $i --trim_gappy 0.9 ; done

for i in *_trimmed.aln ; do mv $i ${i%_trimmed.aln}.aln.trim ; done
```

Total trimmed AA alignments: 11635

---

12. Final tree estimation: homologs

Working directory:

`/OrthoFinder/orthogroup_Homologs_alignments/`

For homologs with <= 500 sequences, use IQ-Tree:

```
iqtree -s INPUT -pre PREFIX -st AA -nt AUTO -m TEST --runs 5 -bb 1000 -wbt > PREFIX.e
rr
```

```
for f in `cat alignment-below200.txt` ; do python3 write_IQ-TREE_AA_tree_PBS_scripts.
py $f tiny12core > ${f%.aln.trim}.iqtree.pbs ; done

sed -i 's/aln.trim/iqtree.pbs/g' alignment-below200.txt

split -l 400 -d alignment-below200.txt alignment-below200-

for f in `cat alignment-below200-00` ; do qsub ${f} ; done
```

Progress of chunks

Chunk	Node	Progress
below500	med12core	done

00	tiny12core	done
01	tiny16core	done
02	tiny16core	done
03	q06h32c	done
04	q06h32c	done
05	tiny16core	done
06	q06h32c	done
07	q06h32c	done
08	tiny16core	done
09	tiny16core	done
10	q06h32c	done
11	tiny12core	done
12	tiny16core	done
13	tiny12core	done
14	tiny16core	done
15	q06h32c	done
16	q06h32c	done
17	tiny12core	done
18	tiny16core	done
19	tiny12core	done
20	tiny12core	done
21	tiny16core	done
22	tiny12core	done
23	tiny16core	done

24	tiny12core	done
25	tiny16core	done
26	tiny12core	done
27	tiny12core	done
28	tiny16core	done

Check for incomplete tree searches and continue them:

```
grep -L "Analysis results written to" OG00*.log
```

The following homologs had IQ-Tree errors: `OG0000209_1rr_1`

For homologs with >500 sequences, use FastTree:

```
FastTree -spr 4 -mlacc 2 -slownni -wag -gamma INPUT > OUTPUT
```

```
for f in `cat alignment-above200.txt` ; do python3 write_fasttree_PBS_scripts.py $f -
w 4 -q tiny12core > ${f%.aln.trim}.fasttree.pbs ; done

sed -i 's/aln.trim/fasttree.pbs/g' alignment-above200.txt

for f in `cat alignment-above200.txt` ; do qsub ${f} ; done
```

---

13. Species tree estimation - example commands

IQ-TREE2 with SRH test to remove bad partitions

```
iqtree -s orthoRT-AA.concat.fasta -pre orthoRT-AA.symtest -st AA -bb 10000 -alrt 1000
0 -m TEST -msub nuclear -spp orthoRT-AA.concat.model --symtest-remove-bad -nt AUTO >>
 orthoRT-AA.symtest.err
```

Gather kept partition names from output and use the same set of gene trees as input to ASTRAL

"orthoRT-AA.symtest.bad.nex" will list the kept partitions

ASTRAL with gene trees that passed SRH test

```
java -jar astral.5.7.3.jar -i AA.symtest.trees -o AA.symtest.astral.tree
```

IQ-TREE2 using the PMSF model

First, create a guide tree using the LG+F+G model

```
iqtree -s orthoRT-AA.top-PI.top-Taxa.concat.fasta -pre orthoRT-AA.pmsf-guide -st AA -
m LG+F+G -nt AUTO >> orthoRT-AA.pmsf-guide.err
```

Second, use the guide tree to run the PMSF model using LG+C20+F+G

```
iqtree -s orthoRT-AA.top-PI.top-Taxa.concat.fasta -pre orthoRT-AA.pmsf -st AA -bb 100
00 -m LG+C20+F+G -nt 24 -ft orthoRT-AA.pmsf-guide.treefile >> orthoRT-AA.pmsf.err
```

---

14. Concordance factor analyses - example commands

Gene and site concordance factors (gCF and sCF)

```
iqtree -t orthoRT-AA.pmsf.treefile --gcf orthoRT-AA.trees --scf 1000 -s orthoRT-AA.co
ncat.fasta -p orthoRT-AA.concat.model --prefix orthoRT-AA.pmsf.treefile
```

Quartet concordance factors (QC)

AA datasets:

```
python3 quartet_sampling.py --tree orthoRT-AA.pmsf.rooted.treefile --align orthoRT-AA
.concat.phy --reps 500 --threads 8 --lnlike 2 --genetrees orthoRT-AA.concat.model --r
esult-prefix RESULT --data-type amino --engine raxml --engine-exec raxmlHPC-PTHREADS-
SSE3 --engine-model PROTGAMMALGF
```

CDS datasets:

```
python3 quartet_sampling.py --tree orthoRT-CDS-pos12.symtest.rooted.treefile --align
orthoRT-CDS-pos12.concat.phy --reps 500 --threads 8 --lnlike 2 --result-prefix RESULT
 --data-type nuc --genetrees orthoRT-CDS-pos12.concat.model --engine raxml --engine-exe
```

---

15. Get dataset summary statistics

Calculate basic summary statistics using AMAS

```
AMAS.py summary -i *.aln.trim -f fasta -d aa -o orthoRT-AA.summary.txt -c 12
```

Calculate GC content for each codon position across genes and taxa using AMAS

```
for i in *.pal2nal.trim ; do extract_codon_alignment --alignedCDS $i --codonPoses 1 --outAln outDir/$${%i.pal2nal.trim}.pos1.pal2nal.trim ; done

for i in *.pos1.pal2nal.trim ; do AMAS.py summary -s -c 12 -i $i -f fasta -d dna ; done

for i in `cat taxon-names.txt` ; do grep -h $i *-summary.txt > $i.summary.txt ; done

for i in *.summary.txt ; do awk '{print $2,$7}' $i > ${i%.summary.txt}.pos1.gc-content.txt ; done

cat *.gc-content.txt > all-taxa.pos1.gc-content.txt
```

Calculate Relative Composition Variability (RCV) using PhyKit

```
conda activate phykit

for i in *.treefile ; do sat=$(phykit saturation -a ${i%.treefile}.pal2nal.trim -t $i) ; rcv=$(phykit rcv ${i%.treefile}.pal2nal.trim) ; echo -e "${i%.treefile}\t$sat\t$rcv" >> saturation-rcv-output.txt ; done
```

Calculate Degree Violation of Molecular Clock (DVMC) and Robinson-Foulds distances from PMSF tree using PhyKit

```
conda activate phykit

for i in *.treefile ; do dvmc=$(phykit dvmc -t $i -r ../root-taxa.txt) ; tl=$(phykit tree_len $i) ; rf=$(phykit rf orthoRT-AA.pmsf.treefile $i) ; echo -e "${i%.treefile}\t$dvmc\t$tl\t$rf" >> orthoRT-AA.clock-genes.output.txt ; done
```

17. Divergence time estimation using MCMCtree

Using the AA dataset. Selected 74 most clock-like (lowest DVMC) and similar (RF similarity to PMSF

species tree) genes.

Use PartitionFinder in IQ-Tree to merge partitions

```
iqtree -s selected-genes.fasta -p selected-genes.model -st AA -m TESTMERGEONLY -T AUT
O -mset JTT,JTTDCMut,LG,WAG -rcluster 10
```

Merged 74 partitions into 19 partitions

"selected-genes.model.best_scheme"

Split the original alignment into the chosen partition scheme (i.e. merge partitions together and output new phylip files for each) --> orthoRT-AA.pmsf.clock-selected.merged.phy

Fossil calibrations:

Node	Calibration
Root	B(0.75,1.18)
Porosira glacialis	L(0.09,0.1,1)
Bacterosira constricta	L(0.0835,0.1,1)
Shionodiscus	L(0.0615,0.1,2)
Cyclostephanos	L(0.05,0.1,1)

Change the following lines in the `mcmctree_run1.ctl` file:

- ndata = 19
- seqtype = 2
- usedata = 3

Run `mcmctree mcmctree_run1.ctl` and delete `out.BV` and `rst` files.

Open the `tmp*.ctl` files and add/change the following lines:

- model = 2
- aaRatefile = jones.dat
- fix_alpha = 0
- alpha = 0.5

- ncatG = 4

Run codeml to generate the Hessian matrix using JTT for each gene.

```
for i in tmp*.ctl ; do codeml $i ; mv rst2 $i.rst2 ; done
```

Combine all *.rst files

```
for f in *.rst2 ; do (cat "${f}"; echo) >> in.BV ; done
```

Edit `mcmctree_run1.ctl` again and set:

- usedata = 2
- RootAge = <1.0
- BDparas = 1 1 0.1
- kappa_gamma = 6 2
- alpha_gamma = 1 1
- rgene_gamma = 2 10 1
- sigma2_gamma = 1 10 1
- print = 1
- burnin = 2500000
- sampfreq = 250
- nsample = 30000

Run MCMCtree

Run0 - no data - `usedata = 0`

Run1 - autocorrelated rates - `clock = 3` `rgene_gamma = 2 20 1`

Run2 - autocorrelated rates - `clock = 3` `rgene_gamma = 2 20 1`

Run3 - autocorrelated rates - `clock = 3` `rgene_gamma = 2 10 1`

Run4 - autocorrelated rates - `clock = 3` `rgene_gamma = 2 10 1`

Run5 - independent rates - `clock = 2` `rgene_gamma = 2 20 1`

Run6 - independent rates - `clock = 2` `rgene_gamma = 2 20 1`