Modified PhyloFisher Pipeline Workflow

Purpose and scope

This document describes a modified PhyloFisher workflow as executed on an HPC system using SLURM. The main deviation from the standard PhyloFisher pipeline is a manual single-gene tree construction step (MAFFT alignment, trimAl trimming, IQ-TREE inference) followed by extensive manual orthology curation with ParaSorter, before re-entering the PhyloFisher workflow to build a concatenated multi-gene matrix and final phylogeny.

All command lines and SLURM batch scripts are reproduced verbatim from the original notes, with only minimal redactions applied to remove potentially sensitive information (usernames, email addresses, project/allocation identifiers, and absolute paths).

Computing environment

Scheduler: SLURM (Uppmax Rackham cluster in the original notes)
Software management: Conda environment + UPPMAX modules (bioinfo-tools)
Key tools: PhyloFisher, MAFFT, trimAl, IQ-TREE2, RAxML-NG, Snakemake, ParaSorter, FigTree

Workflow overview

- Download and extract a standard PhyloFisher dataset (reference).
- Create a project directory and prepare proteomes + metadata mapping.
- Generate PhyloFisher configuration (config.ini).
- Run fisher.py to recover homologs; summarize gene recovery with informant.py.
- Manually construct single-gene alignments and trees (MAFFT -> trimAl -> IQ-TREE) via SLURM.
- Create a PhyloFisher-compatible single-gene tree directory and run forest.py.
- Manually curate orthology/paralogy using ParaSorter; collect edited TSVs.
- Apply curated decisions back to the dataset using apply_to_db.py.
- Select taxa and orthologs (select_taxa.py, select_orthologs.py).
- Prepare final dataset (prep_final_dataset.py) and build the concatenated matrix via Snakemake.
- Infer final phylogeny with IQ-TREE2 (PMSF workflow) and compute bootstrap support with RAxML-NG.

1. Reference dataset download and extraction

Commands used:

```
wget https://ndownloader.figshare.com/files/29093409
tar -xzvf 29093409
```

The archive was extracted into the working directory. Alternative extraction tools (e.g., unzip) can be used depending on the archive type; checking the file type beforehand was recommended in the notes.

2. Project directory and input preparation

Project directory creation and navigation:

```
mkdir Phylofisher_Feb25
cd Phylofisher_Feb25
```

Within the project directory, a folder named "proteomes" was created. All proteome FASTA files and the metadata table were uploaded there. The notes mention that the table would ideally be named input_metadata.tsv to match PhyloFisher conventions.

3. PhyloFisher configuration

The configuration file was created using config.py. The database path (-d) points to the PhyloFisher database directory. The input metadata file (-i) maps proteomes to taxa.

```
config.py
-d
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/PhyloFisherDatab
ase_v1.0/database/
-i
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/proteomes/metada
ta.tsv
```

4. Homolog recovery with fisher.py

PhyloFisher homolog collection was launched via a SLURM batch script that activates the relevant conda environment and runs fisher.py.

Batch script: fisherpy.sh (redacted)

```
#!/bin/bash
#SBATCH -A [REDACTED_PROJECT]  # Ersetze mit deinem Projekt
#SBATCH -M rackham           # Cluster-Name
#SBATCH -p core              # Partition (Knoten-Typ)
#SBATCH -n 15                # Anzahl der CPU-Kerne
#SBATCH -t 2-12:00:00        # Laufzeit: 2 Tage, 12 Stunden
#SBATCH -J phylofisher_job   # Job-Name
#SBATCH -o phylofisher_out.log  # Standard-Output
#SBATCH -e phylofisher_err.log  # Error-Output
#SBATCH --mail-user=[REDACTED_EMAIL]  # Falls du eine Benachrichtigung
willst
```

```
#SBATCH --mail-type=END,FAIL               # Benachrichtigung bei
Beenden/Fehler

# Lade Module (falls benötigt)
module load conda
conda activate /proj/[REDACTED_PROJECT]/conda/envs/fisher # Falls deine
Conda-Umgebung "fisher" heißt

# Wechsle in das Arbeitsverzeichnis
cd /proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25

# Starte dein PhyloFisher-Kommando
fisher.py
```

Submission command:

```
sbatch fisherpy.sh
```

5. Gene recovery statistics with informant.py

After fisher.py completed, informant.py was used to generate per-taxon and per-gene summary statistics in the fisher output directory. This produces an informant_stats directory containing TSV files (e.g., gene presence/absence).

```
informant.py -i fisher_out_Mar.11.2025
```

At this stage it is possible to exclude taxa or genes from downstream single-gene tree inference, but the notes indicate this is possible yet not recommended.

6. Manual single-gene tree inference (custom step)

Instead of relying on the default PhyloFisher single-gene tree constructor, single-gene trees were generated manually. A dedicated directory structure was created to store homolog FASTAs, MAFFT alignments, trimmed alignments, and IQ-TREE outputs.

Directory creation:

```
mkdir single_gene_trees
cd single_gene_trees
mkdir homologs
mkdir mafft
mkdir trimAl
mkdir iqtree
```

Link homolog FASTA files into the homologs directory:

```
ln -s
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/working_dataset_
constructor_out_Mar.12.2025/*.fas .
```

Sanity checks (listing files and symlink targets):

```
ls
ls -ltr
```

Batch script for MAFFT alignment, trimAl trimming, and IQ-TREE inference: mafft_trimAl_iqtree.sh (redacted)

```
#!/bin/bash
#SBATCH -A [REDACTED_PROJECT]
#SBATCH -M rackham
#SBATCH -p core
#SBATCH -n 8
#SBATCH -t 3-00:00:00
#SBATCH --mail-user=[REDACTED_EMAIL]
#SBATCH --mail-type=FAIL
#SBATCH -J mafft_trimAl_iqtree

module load bioinfo-tools
module load MAFFT/7.407
module load trimAl/1.4.1
module load iqtree/2.2.2.6-omp-mpi



fasta=$1

echo "This job processed the input file: $fasta"

#first we do alignment. Note I use these settings to avoid problems with
possibly badly translated regions or introns, perhaps you will not need
them
mafft --unalignlevel 0.6 --adjustdirectionaccurately --thread 2 --reorder -
-maxiterate 1000 --globalpair $fasta > ../mafft/${fasta%.fas}.Mafft-
ginsi.fa

#then we trim our alignments
trimal -in ../mafft/${fasta%.fas}.Mafft-ginsi.fa -out
../trimAl/${fasta%.fas}.final -gt 0.1 -fasta

#finally we run our trees
iqtree2 -s  ../trimAl/${fasta%.fas}.final --prefix ${fasta%.fas} -B 1000 -T
5 -bnni -m LG4X+G

#now will will change the name of the .treefiles to reflect the naming
scheme phylofisher wants, I know we didn't use raxml, but this will let us
feed them back into the pipeline
mv ${fasta%.fas}.treefile ${fasta%.fas}.iqtree.raxml.support

#move the files we want to the correct folder, remove the files we don't
really need
#note if you want to keep these files, you can move them like the first two
files are moved
mv ${fasta%.fas}.iqtree.raxml.support ../iqtree/.
mv ${fasta%.fas}.log ../iqtree/.

rm ${fasta%.fas}.bionj
rm ${fasta%.fas}.ckp.gz
rm ${fasta%.fas}.contree
rm ${fasta%.fas}.iqtree
rm ${fasta%.fas}.mldist
rm ${fasta%.fas}.splits.nex
rm ${fasta%.fas}.ufboot
```
Test run on a single gene (example):

```
sbatch mafft_trimAl_iqtree.sh AGB1.fas
```
Run for all genes:

```
for i in *.fas; do sbatch mafft_trimAl_iqtree.sh $i; done
```
Expected outputs after completion:

- homologs/: symlinked *.fas gene files and SLURM log outputs
- mafft/: *.Mafft-ginsi.fa alignments
- trimAl/: *.final trimmed alignments
- iqtree/: *.iqtree.raxml.support trees and *.log files

7. Creating a PhyloFisher-compatible manual SGT directory

To continue with PhyloFisher, a directory matching the expected output structure of the single-gene tree constructor was created. Required metadata files were copied into this directory to satisfy PhyloFisher's expectations.

```
mkdir manual_sgt_constructor_out_Mar.17.2025
cd manual_sgt_constructor_out_Mar.17.2025
```
Copy required PhyloFisher database and input metadata files into the directory ('.' means current directory):

```
cp
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/PhyloFisherDatab
ase_v1.0/database/metadata.tsv .
cp
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/PhyloFisherDatab
ase_v1.0/database/tree_colors.tsv .
cp
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/proteomes/input_
metadata.tsv .
```
Create trees/ and link trimmed alignments and support trees into it:

```
mkdir trees
cd trees
ln -s
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/single_gen_trees
/trimAl/*.final .
ln -s
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/single_gen_trees
/iqtree/*.iqtree.raxml.support .
```

8. Forest step and manual gene-tree curation

An interactive session was opened, the conda environment activated, and forest.py executed:

```
interactive -A [REDACTED_PROJECT] -M rackham -p core -n 4 -t 12:00:00
module load conda
conda activate /proj/[REDACTED_PROJECT]/conda/envs/fisher
forest.py -i manual_sgt_constructor_out_Mar.17.2025/trees
```

The forest output directory (forest_out_DATE) was used as input for ParaSorter, which allows inspection of per-gene trees across taxa, including identification of orthologs vs paralogs. IQ-TREE outputs were visualized in FigTree; trimmed alignments from trimAl were also inspected. All ~240 single-gene trees were reviewed and compared.

9. Applying ParaSorter curation back to the dataset

After manual inspection and editing, the curated TSV files produced by ParaSorter were collected into a dedicated directory and applied to the PhyloFisher run.

```
ssh -X [REDACTED_EMAIL]
cd /proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25
```

Apply curated TSVs (interactive or batch). Batch script used due to runtime: apply.sh (redacted)

```
#!/bin/bash
#SBATCH -A [REDACTED_PROJECT]  # Ersetze mit deinem Projekt
#SBATCH -M rackham          # Cluster-Name
#SBATCH -p core             # Partition (Knoten-Typ)
#SBATCH -n 15               # Anzahl der CPU-Kerne
#SBATCH -t 2-12:00:00       # Laufzeit: 2 Tage, 12 Stunden
#SBATCH -J phylofisher_job_apply  # Job-Name
#SBATCH -o phylofisher_out_apply.log  # Standard-Output
#SBATCH -e phylofisher_err_apply.log  # Error-Output
#SBATCH --mail-user=[REDACTED_EMAIL]  # Falls du eine Benachrichtigung
willst
#SBATCH --mail-type=END,FAIL               # Benachrichtigung bei
Beenden/Fehler

# Lade Module (falls benötigt)
module load conda
conda activate /proj/[REDACTED_PROJECT]/conda/envs/fisher # Falls deine
Conda-Umgebung "fisher" heißt

# Wechsle in das Arbeitsverzeichnis
cd /proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25

# Starte dein PhyloFisher-Kommando
apply_to_db.py -i parasorter_out_April.1.2025 -fi fisher_out_Mar.11.2025
```

Submission command:

```
sbatch apply.sh
```

10. Taxon and gene selection; final dataset preparation

Taxa and ortholog selection was performed using PhyloFisher helper scripts. These generate editable TSV files where taxa/genes are included or excluded using Yes/No values.

```
select_taxa.py
select_orthologs.py
```

The notes mention permission-related error messages during these steps; however, the required TSV outputs were still produced and used.

Final dataset preparation:

```
prep_final_dataset.py
```

This step generates summary PDFs (gene composition and taxa composition) and creates a prep_final_dataset_out_DATE directory used for matrix construction.

11. Concatenated matrix construction with Snakemake

The concatenated supermatrix was generated via the matrix_constructor.smk Snakemake workflow. The run was submitted via a SLURM batch script: run_snakemake.sh (redacted).

```
#!/bin/bash
#SBATCH -A [REDACTED_PROJECT]  # Ersetze mit deinem Projekt
#SBATCH -M rackham          # Cluster-Name
#SBATCH -p core             # Partition (Knoten-Typ)
#SBATCH -n 15               # Anzahl der CPU-Kerne
#SBATCH -t 2-12:00:00       # Laufzeit: 2 Tage, 12 Stunden
#SBATCH -J matrix_constructor_job  # Job-Name
#SBATCH -o matrix_constructor_out.log  # Standard-Output
#SBATCH -e matrix_constructor_err.log  # Error-Output
#SBATCH --mail-user=[REDACTED_EMAIL]  # Falls du eine Benachrichtigung
willst
#SBATCH --mail-type=END,FAIL               # Benachrichtigung bei
Beenden/Fehler

# Lade Module (falls benötigt)
module load conda
conda activate /proj/[REDACTED_PROJECT]/conda/envs/fisher  # Falls deine
Conda-Umgebung "fisher" heißt

# Wechsle in das Arbeitsverzeichnis
cd /proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25

snakemake -s
/crex/proj/[REDACTED_PROJECT]/conda/envs/fisher/bin/matrix_constructor.smk
\
--cores 1 \
--config \
out_dir=/crex/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/mat
rix_constructor_out_1 \
in_dir=/crex/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/prep
_final_dataset_out_1 \
in_format=fasta \
out_format=fasta \
concatenation_only=False \
trimal_gt=0.8 \
genes="MAT,SCSB,GDI,FTSJ1,ATG2,AMP2B,RPS16,AP3S1,PSMA-H,DHSB3,RPTOR,PSMA-
E,PYGB,PSMB-M,UBE12,DNAI2,RPL33,TMS,PACE2-A,CPN60,SPTC2,DHSA1,RPO-
C,RPAC1,CCDC113,VATB,CALR,RPS10,COPS6,RPS12,CCT-Z,RPS8,NDUFV2-
MITO,ODPB,RPL3,RPS6,RPS15,GLGB2,EFG-MITO,RPS23,COPE,IMB1,PSMB-
N,GDI2,WBSCR22,RPS11,ORF2,NSF1-G,MCM-
B,RPL21,SYGM1,RPL2,CTP,PACE2C,EFTUD1,KDELR2,IF2B,RPS2,GCST,METTL1,PURA,RPL1
9,NSF1-K,RPL31,PSMA-C,PIK3C3,RPL5,CCT-
T,SPTLC1,IF2P,RPL43,VATE,L10A,NMD3,ODBB,AP3M1,DIMT1L,ATP6,COP-
BETA,YKT6,GNL2,RPO-A,CCT-
B,GSS,RPL44,RPS17,ARP2,CRNL1,NAA15,RPL35,GNB2L,RPF1,RPL32,HYOU1,BTUB,ATSAR2
```

```
,RPS4,CLAT,VATC,RPL9,PSD11,OPLAH,SAP40,VPS26B,TOPO1,PSMD,RPS18,WRS,SND1,PPX
2,XPB,ATP6V0A1,ARPC1,EIF3C,MRA1,PACE5,NSF1-H,IFT88,DRG2,NSA2,MCM-
A,RPL14E,PSMA-F,PGM2,ALIS1,IFT57,NSF1-I,PSD7,H2A,WD66,BAT1,RRAGD,PSMA-
J,NFS1-MITO,RAD51A,SUCA,SEC23,CRFG,IPO4,PSMB-
L,VATA,WD,RPL13A,RF1,FAM,VPS18,CCDC37,ODBA,RHEB,TRS,ODPA2,CCT-
D,NOP5A,RPPO,NSF1-M,RPL7A,CCDC65,AGB1,MCM-C,RAN,ADK2,EIF3I,NSF1-
C,EIF4A3,GRC5,RPL13E,PSMA-
G,GTUB,CDK5,RPN1B,RPS26,PSMD12,STXBP1,GLCN,S15A,PPP2R3,RPL20,IF6,CCT-
E,IMP4,APBLC,SCO1-MITO,ATP6V0D1,VAPA,SRP54,AP4M,ALG11,CCT-
A,IPO5,C3H4,CCDC40,RPL24A,IFT46,RPL11,PSMB-K,CCT-
N,ARPC4,CAPZ,NDF1,HSP70MT,S15P,PPP2R5C,RPL30,RPL17,UBA3,GMPP3,ODO2A,MTHFR,C
S,RPS3,CORO1C,RPO-B,RPS20,PSMA-B,CC1,AR21,AP4S1,IF2G,RPS5,PSMA-
A,RPL15,RPL12,ARP3,MCM-E,CCT-G,NSF1-L,DNAL1,TM9SF1,NSF1-
J,MTLPD2,HMT1,SRA,COPG2,SF3B2,RPL4B,PSMD6,MCM-D,AGX,VBP1,RICTOR" \
--forceall \
-- \
/crex/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_cons
tructor_out_1/matrix.fas \
/crex/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_cons
tructor_out_1/indices.tsv \
/crex/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_cons
tructor_out_1/matrix_constructor_stats.tsv \
/crex/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_cons
tructor_out_1/occupancy.tsv
```

Submission command:

```
sbatch run_snakemake.sh
```

12. Final phylogenomic inference and support values

After the supermatrix was created, IQ-TREE2 was used to infer a phylogenomic tree using a PMSF strategy (guide tree under C20, then C60 with site frequencies). A RealBootstrap computation was performed, and final support values were mapped using RAxML-NG.

Batch script: iqtree1.sh (redacted)

```
#!/bin/bash
#SBATCH -A [REDACTED_PROJECT]
#SBATCH -M rackham
#SBATCH -p core
#SBATCH -n 16
#SBATCH -t 9-00:00:00
#SBATCH -J IQTree1
#SBATCH -o IQTree1.out
#SBATCH -e IQTree1.err
#SBATCH --mail-user=[REDACTED_EMAIL]
#SBATCH --mail-type=END,FAIL
#SBATCH --mem=160G
#SBATCH --exclusive

set -e

# Erstelle einen neuen Ordner für die Ausgaben
WORK_DIR="/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/IQTree
1_$(date +%F)"
mkdir -p "$WORK_DIR"
```

```
cd "$WORK_DIR"

# Lade die benötigten Module
module load bioinfo-tools
module load iqtree/2.2.2.6-omp-mpi  # Je nach Version anpassen

# 1. Schritt: IQTree2 Baumaufbau (Modell LG+C20+F+G)
echo "Starte Baumaufbau mit IQTree2 (Modell LG+C20+F+G)..."
iqtree2 -T 16 \
  -m LG+C20+F+G \
  -s
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_construct
or_out_1/matrix.fas \
  -pre "$WORK_DIR/PF.tutorial.LGC20GF"

# 2. Schritt: IQTree2 Baumaufbau (Modell LG+C60+F+G)
echo "Starte Baumaufbau mit IQTree2 (Modell LG+C60+F+G)..."
iqtree2 -T 16 \
  -ft "$WORK_DIR/PF.tutorial.LGC20GF.treefile" \
  -m LG+C60+F+G \
  -s
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_construct
or_out_1/matrix.fas \
  -pre "$WORK_DIR/PF.tutorial.LGC60GCF-PMSF"

# 3. Schritt: RealBootstrap Berechnung
echo "Starte RealBootstrap Berechnung..."
mkdir -p "$WORK_DIR/REALBS"
cd "$WORK_DIR/REALBS"
iqtree2 -T 16 \
  -m LG+C60+F+G \
  -s
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_construct
or_out_1/matrix.fas \
  -fs "$WORK_DIR/PF.tutorial.LGC60GCF-PMSF.sitefreq" \
  -pre "$WORK_DIR/PF.tutorial.LGC60GCF-PMSF.realbs_100" \
  -mem 160G \
  --bonly 100

# 4. Schritt: RAxML Bootstrapping
echo "Starte RAxML Bootstrapping..."

module load bioinfo-tools
module load RAxML-NG/1.1.0

raxml-ng --support --threads 16 \
  --tree "$WORK_DIR/PF.tutorial.LGC60GCF-PMSF.treefile" \
  --bs-trees "$WORK_DIR/PF.tutorial.LGC60GCF-PMSF.realbs_1000.boottrees" \
  --msa
/proj/[REDACTED_PROJECT]/[REDACTED_USER]/Phylofisher_Feb25/matrix_construct
or_out_1/matrix.fas \
  --model PROTGAMMALG \
  --prefix "$WORK_DIR/PF.LGC60GCF.PMSF.support.1000"

echo "Workflow abgeschlossen!"
```
Job monitoring command:

```
scontrol show job [JOBID]
```

Appendix: Redaction policy applied

To reduce the risk of exposing personal or institution-specific information, the following items were redacted where they appeared in commands and scripts:
- Email addresses
- SLURM account/allocation identifiers
- Usernames in SSH commands and directory paths
- Absolute paths containing project/user-specific directories

No computational parameters, tool options, or logic were changed.