

PHYLOGENOMICS WORKSHOP

This phylogenomics tutorial is divided into 3 major sections. The first section deals with identification of orthologs from closely related plasmodium species. Second section is about multiple sequence alignment and construction of super matrix. The final section will show simple phylogenetic tree construction using the Maximum Likelihood and Bayesian analyses.

In this workshop, we will first use ‘OrthoMCL’ program to detect orthologous sequences from the Plasmodium genomes (predicted proteins), choose only single copy orthologs, perform alignment using ‘MUSCLE’ program, trim the alignments using ‘trimAl’, concatenate the alignments using ‘FASconCAT’, build trees using Maximum Likelihood (RAxML) and Bayesian analyses (MrBayes).

PREREQUISITES

To run OrthoMCL program to detect orthologs, we need MySQL database. Since we do not have a dedicated server providing MySQL database, we will provide a Linux image file with pre-installed MySQL and OrthoMCL software. Once a proper environment is setup, users can load this image file and run Linux within the existing operating system (host) without changing anything. In this guide we will describe how to setup this environment, load the image and run the Linux machine.

SETTING UP VIRTUALBOX

To use the Linux image that we provided we need to set up a proper environment. This requires installation of ‘Oracle VM VirtualBox’ software. Although these instructions are for Windows, similar steps can also be used for Mac. **This is already been pre-installed in lab computers**. Just type in the name of the program (VirtualBox) in the search bar and open it. If you want to set it up on your personal computer then:

1. Download ‘Oracle VM VirtualBox’ from the product webpage. There are several versions for different platforms. Select the one you need: <http://www.oracle.com/technetwork/server-storage/virtualbox/downloads/index.html>
2. After you have downloaded and installed, you should execute the program you just installed.

NOTE: This step will reconfigure your network, which will cause you to disconnect the network and then reconnect.

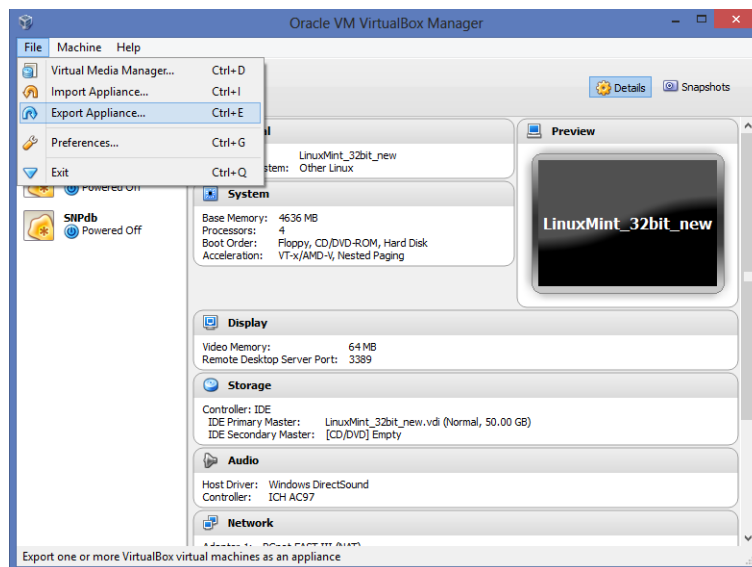
LOADING LINUX IMAGE

To start using the Linux machine virtually (running it as a program within your operating system) you need to first import the appliance. If you haven’t already downloaded the Linux image file, you can get it from here

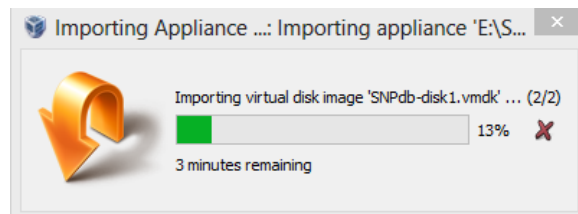
<https://wiki.itap.purdue.edu/display/BioCoreFacility/Phylogenomics+workshop>


Follow the steps along with the screenshots to complete this setup.

1. Open the 'Oracle VM VirtualBox Manager', click on 'File', 'Import Appliance'. Alternatively, you can just double click on the downloaded file to begin import.



2. Click on 'Open Appliance' and browse for the Linux image file (workshop.ova) you downloaded. Select the file and click 'Next' followed by import. It will take few minutes to import the image



3. Once done, click on  (start), Linux will start up in new window. You can make it full screen or let it run as it is. This Linux version has preinstalled software packages required for today's workshop. You can also save this image and use it anytime in future for other data as well.

NOTE: Once you setup the Virtual Box Linux machine **DO NOT SHUT DOWN or LOG OFF your computer**. If you do, then you have to start all over again, since it will wipe out the Linux installation along with the data stored in it.

PDF version of this manual can be downloaded from this link:
http://web.ics.purdue.edu/~aseethar/phylogenomics_workshop.pdf

IDENTIFICATION OF ORTHOLOGS

Once you have completed the previous steps, you can now start using the Linux machine for detecting orthologs. For detecting orthologs we will use OrthoMCL software package (Fischer et al. 2011).

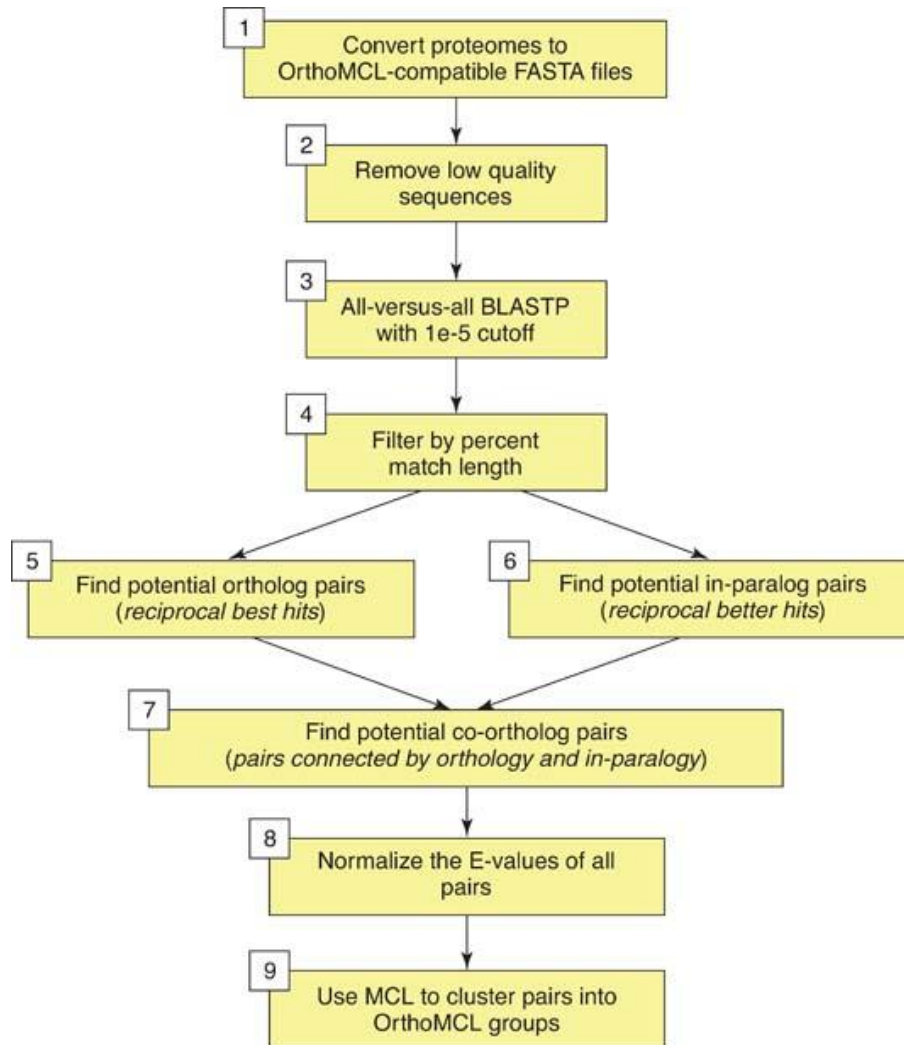


Fig 1: Overview of the OrthoMCL algorithm (Fischer et al. 2011)

There are four major steps for detecting orthologs:

1. Prepare your FASTA files (protein sequences for each genome, separately) [box 1,2 in figure]
2. Run all-versus-all BLASTP (using NCBI BLAST) [box 3, in figure]
3. Parse the BLASTP results and load it into the relational database and running the OrthoMCL software to find significant pairs of proteins [box 4-8, in figure]
4. Use the MCL software to cluster the pairs into groups [box 9, in figure]

Once clustered, we will use custom scripts to extract single copy orthologs and proceed with phylogenetic tree construction.

GETTING DATA

We will begin by downloading the data. Open the terminal in your Linux VM and run the following commands (**VM already has these files, skip next 2 steps**):

```
wget http://web.ics.purdue.edu/~aseethar/workshop_files.tar.gz    # Download
tar -xvzf workshop_files.tar.gz                                # Extract tar gzipped file
cd workshop_files                                             # change directory
```

Within the `workshop_files`, you will find 2 directories and seven plasmodium whole genome protein datasets. `Scripts` directory contains required scripts and `pre-computed_results` directory, contains result files for each step. We will work with FASTA files now. The files name will have information such as genus, species and strain name. To open a file,

```
less PbergheiANKA.fasta                                       # view file contents, press 'q' to quit
```

If you look at the `fasta` headers, you can see that there is lot of additional information for each sequence. Since, we will identify the orthologs based on similarity, we won't use any of these information. Also, long sequence names might interfere with the downstream analyses. So we will clean all the sequences first.

```
mkdir original                                               # make a new directory to keep original files
mv *.fasta ./original/                                       # move all fasta files to the new directory
mkdir complaintFasta                                         # new directory for processed files
cd original                                                  # change directory
```

CLEANING SEQUENCES

We will use '`orthomclAdjustFasta`' to clean our sequences. In the following commands, the first option (eg., `Pberghe`) is used as taxonomic identifier, second option (eg. `PbergheiANKA_Proteins.fasta`) as the input file name and last option (eg. `1`) as the field to be used as sequence identifier from the input file.

```
orthomclAdjustFasta Pberghe PbergheiANKA.fasta 1
```

We will do this for all the genomes:

```
orthomclAdjustFasta Pchabau Pchabaudichabaudi.fasta 1
```

```
orthomclAdjustFasta Pcynomo PcynomolgiB.fasta 1
```

```
orthomclAdjustFasta Pfalcip Pfalciparum3D7.fasta 1
```

```
orthomclAdjustFasta Pknowle PknowlesiH.fasta 1
```

```
orthomclAdjustFasta Pvivaxs PvivaxSaI1.fasta 1
```

```
orthomclAdjustFasta Pyoelii PyoeliiyoeliiYM.fasta 1
```

This will generate specifically formatted fasta file that can be used with OrthoMCL software. We will move these files to another directory (complaintFasta).

```
mv ???????.fasta ../complaintFasta # move new files to complaintFasta
cd ../ # change one directory up
```

FILTERING SEQUENCES

Since the predicted proteins will have proteins of all sizes, we need to perform a filtering step where we remove all proteins that are shorter than a specific length, so that they won't interfere with the OrthoMCL analyses.

```
orthomclFilterFasta complaintFasta 10 20
```

Here, 10 is the minimum length for protein to keep and 20 is the maximum allowed stop codons in the sequences. This command will generate 2 files: `goodProteins.fasta`, containing all proteins that passed the filtering and `poorProteins.fasta`, containing all rejects. You can view those files by opening it using `less` command.

ALL VS. ALL BLAST

The next step is to perform all vs. all BLAST. This step will be performed using the NCBI-BLAST program (Altschul et al. 1990) with `goodProteins.fasta` as the BLAST database and as query sequences.

First, prepare the BLAST database:

```
makeblastdb -in goodProteins.fasta -dbtype prot -parse_seqids -out goodProteins.fasta
```

Options `-dbtype` will specify the input sequences as proteins, `-parse_seqids` will allow extracting the sequences from the database and `-out` specifies the blast database name.

This will create a database (4 additional files). For performing BLAST:

```
blastp -db goodProteins.fasta -query goodProteins.fasta -outfmt 6 -out
blastresults.tsv -num_threads 8 # DON'T RUN THIS!
```

Here, `-db` specifies the database that will be used (same database that was generated in the previous step), `-query`, specifies the input query sequences for performing blast, `-outfmt`, selects option 6 which is tab separated output format for blast results, `-out` is the BLAST results file name and `-num_threads` refers to total number of processors to be used for BLAST.

This step is very time consuming. For the above data, it took about 10 hours to complete. So for today's workshop we will skip this step and use the pre-computed BLAST results.

```
cp pre-computed_results/blastresults.tsv ./ # Copy blastresults.tsv
```

For future reference, you can follow these steps to perform BLAST on servers. **You don't have to run any of the following steps in this section.**

To transfer the file (`goodProteins.fasta`) using following command

```
scp goodProteins.fasta username@coates.rcac.purdue.edu: # enter password
```

Then use a submission script (jobfiles/step_1.1_blastp.sub) to run BLAST

```
cat step_1.1_blastp.sub # view contents
```

```
#!/bin/bash # bash script header
#PBS -q bioinformatics # bioinformatics queue
#PBS -l walltime=48:00:00 # allot 48 hrs for this job
#PBS -l nodes=1:ppn=8 # 1 node and 8 processors/node
#PBS -N BLAST # job name
cd $PBS_O_WORKDIR # run program from present dir
module use /apps/group/bioinformatics/modules # use modules
module load blast # load BLAST program
makeblastdb -in goodProteins.fasta -dbtype prot -parse_seqids -out
goodProteins.fasta # format database
blastp -db goodProteins.fasta -query goodProteins.fasta -outfmt 6 -out
blastresults.tsv -num_threads 8 # run blastp program
```

```
qsub step_1.1_blastp.sub # submit job
```

PARSE BLAST RESULTS

Next we will use orthomclBlastParser program to convert the tab delimited BLAST results into a format ready for loading into the OrthoMCL schema in the relational database.

```
orthomclBlastParser blastresults.tsv ./complaintFasta/ >> similarSequences.txt
```

This step will also compute the percent match and percent identity of each hit. The resulting file will be uploaded to the relational database.

CONFIGURE DATABASE

Before we upload the similarSequences.txt file to the database, we need to set up the database (**the orthomcl database is already setup in this VM, so these steps are only for your reference -skip next 4 steps**)

```
mysql -u purduepete -p # to get mysql prompt, enter password boilerup!
create database orthomcl; # create a database called 'orthomcl'
show databases; # shows all databases present
exit # exit mysql prompt
```

These steps can also be performed via web interface via <http://localhost/phpmyadmin> , username is root and password is boilerup!

We will also need a configuration file that will specify OrthoMCL how to communicate with the MySQL database. For this we will create a file called mysql.config

```
cp scripts/mysql.config ./ # copy file
```

```
cat mysql.config # view contents
```

```
dbVendor=mysql
dbConnectionString=dbi:mysql:orthomcl:mysql_local_infile=1:localhost
dbLogin=purduepete
dbPassword=boilerup!
similarSequencesTable=SimilarSequences
orthologTable=Ortholog
inParalogTable=InParalog
coOrthologTable=CoOrtholog
interTaxonMatchView=InterTaxonMatch
percentMatchCutoff=50
evaluateExponentCutoff=-5
oracleIndexTblSpc=NONE
```

To specify the structure for the orthomcl database that we just created, we will use the following command:

```
orthomclInstallSchema mysql.config mysql.log
```

Now, the database is ready to up load the similarSequences.txt file

UPLOAD DATA INTO THE DATABASE

```
orthomclLoadBlast mysql.config similarSequences.txt
```

Once uploaded we can call pairs (potential orthologs, co-orthologs and in-paralogs).

```
orthomclPairs mysql.config pairs.log cleanup=no
```

This is a computationally intensive step that finds protein pairs looking in to the BLAST results that was uploaded. This program executes a series of 20 internal steps, each creating an intermediate database table or index. Finally, it populates the three output tables: Orthologs, InParalogs and CoOrthologs.

cleanup=no, will make sure that all intermediary tables in the database are kept so that, if one of those steps fails, you can restart from the step it failed. You can also use options such as yes (not to keep intermediate tables), only (drops table if all steps are successful), all (retains only final 3 tables).

GETTING RESULTS

To get the results back from the database made by orthomclPairs, orthomclDumpPairsFiles command can be used.

```
orthomclDumpPairsFiles mysql.config
```

The output will be a directory (called pairs) and a file (called mclinput). The pairs directory, will contain three files: orthologs.txt, coorthologs.txt, inparalogs.txt. Each of these files describes pair-wise relationships between proteins. They have three columns: Protein 1, Protein 2 and normalized similarity score between them. The mclinput file contains the identical information as the three files in pairs directory but merged as a single file and in a format accepted by the mcl program.

MCL program (Dongen 2000) will be used to cluster the pairs extracted in the previous steps to determine ortholog groups.

```
mcl mclInput --abc -I 1.5 -o groups_1.5.txt
```

Here, --abc refers to the input format (tab delimited, 3 fields format), -I refers to inflation value and -o refers to output file name. Inflation value will determine how tight the clusters will be. It can range from 1 to 6, but most publications use values between 1.2 -1.5 for detecting orthologous groups.

The final step is to name the groups called by mcl program.

```
orthomclMclToGroups OG1.5_ 1000 < groups_1.5.txt > named_groups_1.5.txt
```

Here, OG1.5_ is the prefix we use to name the ortholog group, 1000 is the starting number for the ortholog group and last 2 fields are input and output file name respectively.

FORMATTING RESULTS

Once we have the Group numbers and IDs (named_groups_1.5.txt), we can generate a summary table showing number of genes present in each genome for each ortholog group as well as extract ortholog groups that have single copy gene in each species. We will use 2 custom scripts for this purpose:

```
scripts/CopyNumberGen.sh named_groups_1.5.txt > named_groups_1.5_frequency.txt
```

```
scripts/ExtractSCOs.sh named_groups_1.5_frequency.txt > scos_list.txt
```

The scos_list.txt generated will have all ortholog groups that have only one copy gene in all species. We will use this list to generate a file similar to named_groups_1.5.txt but with only ortholog groups that are present in scos_list.txt

```
cut -f 1 scos_list.txt > ids.txt # cut first field from the file
```

```
while read line; do \  

grep -w "$line" named_groups_1.5.txt; \  

done<ids.txt > named_sco_groups_1.5.txt # filtering group file
```

The same command above to copy/paste:

```
while read line; do grep -w "$line" named_groups_1.5.txt; done<ids.txt >  

named_sco_groups_1.5.txt
```

EXTRACTING SEQUENCES

The final step is to extract the sequences, to save time we will only use first 100 ortholog groups to build tree

```
head -n 100 named_sco_groups_1.5.txt > 100_input_list.txt # take top 100 lines
```

```
scripts/ExtractSeq.sh -o sequences 100_input_list.txt goodProteins.fasta
```


Here, -o is the output folder where sequences will be saved, 100_input_list.txt is the list of ids that will be used to extract sequences and goodProteins.fasta is the database from which the sequences will be extracted. Note: this step requires NCBI-BLAST.

Once we are done, we will tar zip the sequences folder, export it to clusters to continue phylogenomics exercise

```
tar -cvzf sequences.tar.gz sequences          # tar gzip the sequences folder
scp sequences.tar.gz username@coates.rcac.purdue.edu: # copy file to clusters
```

NOTE: Answer “yes” if it asks for security confirmation and enter your password when prompted. Passwords won’t appear on the screen when typed.

MULTIPLE SEQUENCE ALIGNMENT

There are many multiple sequence alignment (MSA) software programs for aligning protein sequences. The choice of the tool to use depends on our input data type. A simple decision tree is shown below for selecting the right one.

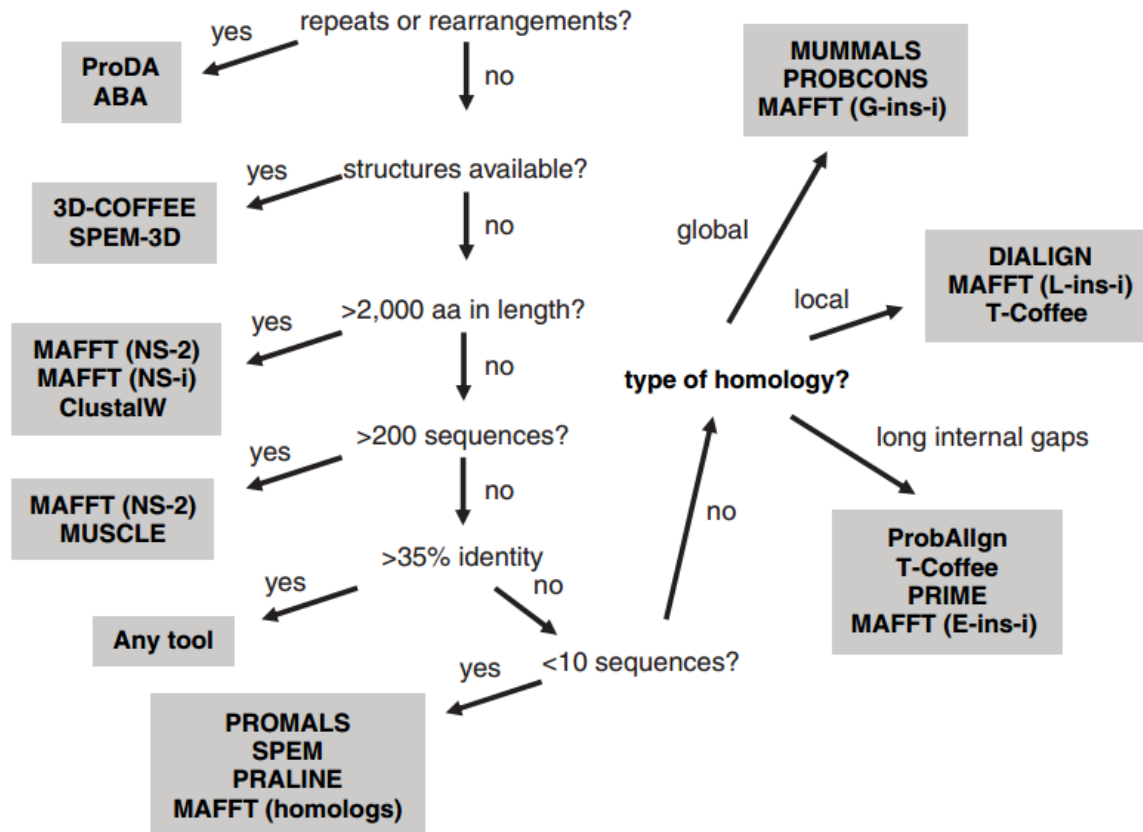


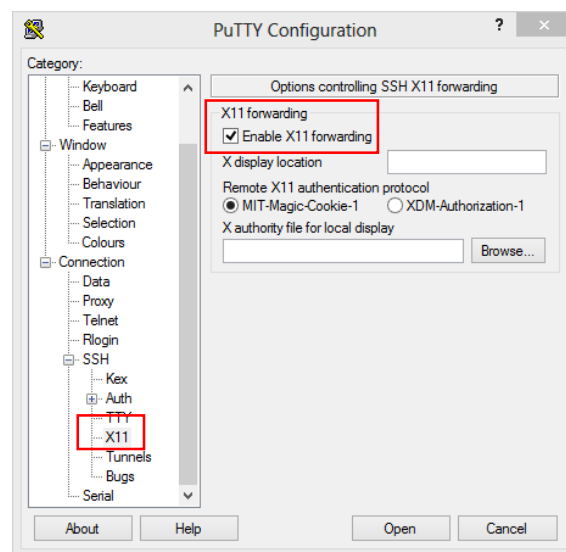
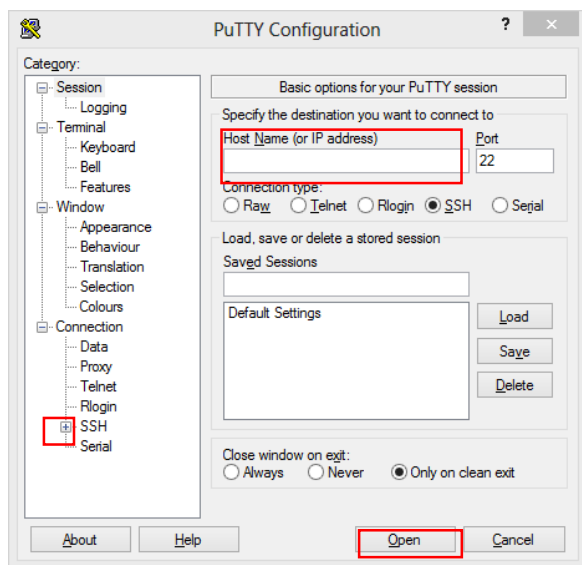
Fig 2: Decision tree for selecting an appropriate MSA tool ((Do & Katoh 2008)

For today’s workshop, we will use MUSCLE for MSA.

GETTING STARTED

Login to the Coates cluster using the Purdue credentials. Search for PuTTY, after clicking on Start/Windows button. Open the “PuTTY” program, you will see “PuTTY configuration” window as shown.

- Type in the host name `coates.rcac.purdue.edu`
- Enable X11: Under connection, click on ‘+’ near SSH, click on X11 under SSH, and check “Enable X11 forwarding”. Go back to the session page by clicking on “Session” on the top.
- If you wish to save this configuration, type a name in “saved sessions” and click on “save”
- Click on start/windows button and search for Xming, click on it and it will start running in the background (you will only see a system tray icon).
- Click “open” and type your password.
- Today’s Coates access is temporary and only meant for use with this workshop
- Limited access, free Coates cluster accounts are provided upon request contact bioinformatics@purdue.edu



DATA EXTRACTION

You should have the `sequences.tar.gz` file in your home directory. We will move this file from home to scratch space, which is ideal for performing large jobs.

```
cd $RCAC_SCRATCH # change from home to scratch
cp -r ~aseethar/phylogenomics ./ # copy additional files required
cp ~/sequences.tar.gz ./phylogenomics # copy sequences from home
cd phylogenomics # change directory
tar -xvzf sequences.tar.gz # extract/unzip the file
```

```
cd sequences # change directory
```

The FASTA sequences in this directory will have headers with following structure:

```
>lcl|Pspecies|PXXX_1234567 unnamed protein product
```

In order to make a super matrix, we need to combine the sequences belonging to each species together. So we will run the cleaning script on these FASTA files as follows:

```
for f in *.fa; do \ # for every file with .fa extension
sed -i 's/>lcl|(\.{7}\)\.\+/>l/g' $f; \ # format the name
done;
```

The same command above to copy/paste:

```
for f in *.fa; do sed -i 's/>lcl|(\.{7}\)\.\+/>l/g' $f; done;
```

This will change the header as:

```
>Pspecies
```

MULTIPLE ALIGNMENT

Now we are ready to run the MUSCLE multiple alignment program (Edgar 2004). A submission file to perform the alignment is already created and placed in jobfiles directory. We will copy this to our workspace directory.

```
cp ../jobfiles/step_2.1_muscle.sub ./ # copy submission file
```

```
cat step_2.1_muscle.sub # view contents
```

```
#!/bin/bash # bash script header
#PBS -q bioinformatics # bioinformatics queue
#PBS -l walltime=10:00:00 # allot 10 hrs for this job
#PBS -l nodes=1:ppn=2 # 1 node and 2 processors/node
#PBS -N MUSCLE # job name
cd $PBS_O_WORKDIR # run program from present dir
module use /apps/group/bioinformatics/modules # use modules
module load muscle # load MUSCLE program
for f in *.fa; # loop thru .fa files
do muscle -in $f -out $f.out; # do multiple alignment
done
```

```
qsub step_2.1_muscle.sub # submit job
```

You will receive a confirmation

```
1234567.coates-adm.rcac.purdue.edu # number is your job ID
```

This will take about 5-10 minutes to complete. You can check the status of your job using qstat command

```
qstat -u username
```

Once completed you will see the following files in the directory:

```
MUSCLE.e1234567 # standard error
MUSCLE.o1234567 # standard out
(100 files with .fa.out extension) # alignment output (FASTA format)
```

We will use all files with .fa.out extension for next steps.

```
cd .. # change to one directory up
mkdir aligned # create a new directory
mv ./sequences/*.out ./aligned/ # move aligned files to the new directory
cd aligned # change to new directory
```

TRIMMING ALIGNMENT

To trim the alignment (to make all the aligned sequence of same length), we will use trimAl (Capella-Gutierrez et al. 2009) tool

```
module use /apps/group/bioinformatics/modules # use bioinformatics modules
module load trimal # load trimal software
```

You can find information about this tool by entering `trimal -h` on command line, here we will just use with following options

```
trimal -in alignment_file -out output_file.phy -phylip -automated1 # DON'T RUN THIS!
```

To run trimAl on all files, we will use the loop again:

```
for f in *.fa.out; do \ # for every file in the directory
g=$(echo $f | sed 's/\.fa.out$//g');\ # generate output name
trimal -in $f -out $g.phy -phylip -automated1;\ # outputs phylip trimmed file
done
```

The same command above to copy/paste:

```
for f in *.fa.out; do g=$(echo $f | sed 's/\.fa.out$//g'); trimal -in $f -out $g.phy -
phylip -automated1; done
```

Now we have trimmed alignment in PHYLIP format. This will be used to concatenate and generate a supermatrix using FASconCAT perl script (Kuck & Meusemann 2010)

```
mkdir ../trimmed # make a new directory
mv *.phy ../trimmed # move all .phy files
cd ../trimmed # change directory to trimmed
module load FASconCAT # load FASconCAT software
```

FASconCAT_v1.0.pl

This will bring up the options menu, the top part shows the options to configure, while the bottom part shows the selected options. Below this you will see a prompt where you can enter commands to configure these sections. We need to change few things: First press 'i' to change 'Supermatrix + ALL info' to 'YES', second, press 'p + Enter' twice to change 'PHYLP' to 'RELAXED' and finally press 's' to start concatenating the alignment files.

You will see 3 output files FcC_info.xls, FcC_smatrix.fas and FcC_smatrix.phy. XLS file is the information about concatenated files and FAS and PHY files are concatenated alignments in fasta and phylip format.

```
mv FcC_* ../                # move results one directory up
cd ../                      # change one directory up
```

Since we also need this super-alignment file in Nexus format as well, we will use readAl script (Capella-Gutierrez et al. 2009)

```
readal -in FcC_smatrix.fas -out FcC_smatrix.nex -nexus

mkdir MrBayes RAxML        # create new directories

mv FcC_smatrix.nex ./MrBayes/  # move input file (NEXUS)

cp FcC_smatrix.phy ./RAxML/    # copy input file (PHYLIP)
```

TESTING MODELS

To test what evolutionary model to use on the trimmed alignment file, we can run ProtTest V3.0 (Darriba et al. 2011). The best evolutionary model is determined by various framework, such as Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Second-Order Akaike framework (AICc) and Decision Theory (DT). Once the model is selected, ML and Bayesian trees can be constructed using that model.

To run ProtTest3.3, we will first copy executables

```
module load prottest

cp -r $PROTTEST_SRC ./      # copy directory

cp FcC_smatrix.phy ./prottest-3.3-20130716  # copy alignment file

cp jobfiles/step_3.1_prottest.sub ./prottest-3.3-20130716 # copy submission file

cd prottest-3.3-20130716    # change directory
```

```
cat step_3.1_protest.sub
```

```
# view contents
```

```
#!/bin/bash # bash script header
#PBS -q bioinformatics # bioinformatics queue
#PBS -l walltime=48:00:00 # allot 48 hrs for this job
#PBS -l nodes=1:ppn=2 # 1 node and 2 processors/node
#PBS -N ProtTest # job name
cd $PBS_O_WORKDIR # run program from present dir
module use /apps/group/bioinformatics/modules # use modules
module load java # load ProtTest program
java -jar prottest-3.3.jar -i FcC_smatrix.phy -o test_results.txt -Dayhoff -WAG -
JTT -VT -Blosum62 -DCMut -all -all-distributions -F -S 1 -AIC -BIC -AICC -DT -
threads 2 # test models
```

```
qsub step_3.1_protest.sub
```

```
# submit job
```

Here,

```
-i Input file
-o Output file
-Dayhoff
-WAG
-JTT
-VT
-Blosum62
-DCMut } Models to test
-all will display a table comparing all model selection frameworks,
-all-distributions Includes testing for parameters such as gamma distributions,
invariant sites and invariant gamma distributions,
includes models with estimated amino acid frequencies,
-F
-AIC
-BIC
-AICC
-DT } selects best models based on different framework
```

It will take ~1 hour to complete. You can check the status of your job using qstat command

```
qstat -u username
```

Once completed you will see the following files in the directory:

```
ProtTest.e1234567 # standard error, number=job id
ProtTest.o1234567 # standard output
test_results.txt # results
```

You can open the test_results.txt using less command and see which model was selected for the data. It will also inform you about what parameters could be used with the model.

Alternatively, if you have a smaller job, you can run GUI version of ProtTest3.3. Within the protest-3.3 directory, execute

```
sh runXProtTestHPC.sh # gui version
cp ../pre-computed_results/ProtTest/test_results.txt ./ # copy results
less test_results.txt # view contents
cd ../ # one directory up
```

A window will popup, showing interface for GUI version of ProtTest. Click on File >> Load Alignment, *browse for FcC_smatrix.phy file* and click open. Window will show you some statistics about the alignment. To perform model testing, click on Analysis >> Compute Likelihood Scores, *make all required selections* and click compute.

PHYLOGENETIC TREE CONSTRUCTION

We will use two different methods to generate phylogenetic tree. First method utilizes maximum likelihood tree reconstruction and bootstrapping using RAXML (Randomized Axelerated Maximum Likelihood) (Stamatakis 2006), which was developed by Alexandros Stamatakis. Second method is based on Bayesian analyses as implemented by MrBayes (Altekar et al. 2004).

MAXIMUM LIKELIHOOD TREE

For RAXML, we will just use standard settings with the ProtTest estimated models and parameters (un-partitioned data, best ML tree estimated from 1000 bootstraps with bootstrap values).

```
cd RAXML
cp ../jobfiles/step_4.1_raxml.sub ./ # copy submission file
cat step_4.1_raxml.sub # view contents
```

```
#!/bin/bash # bash script header
#PBS -q bioinformatics # bioinformatics queue
#PBS -l walltime=48:00:00 # allot 48 hrs for this job
#PBS -l nodes=1:ppn=8 # 1 node and 2 processors/node
#PBS -N RAXML # job name
cd $PBS_O_WORKDIR # run program from present dir
module use /apps/group/bioinformatics/modules # use modules
module load RAXML # load RAXML program
raxmlHPC-PTHREADS-SSE3 -T 8 -p 12345 -f a -s FcC_smatrix.phy -n results.tree -c 4
-m PROTGAMMAIJTTF -x 12345 -N 1000 # run RAXML
```

```
qsub step_4.1_raxml.sub # submit job
```

For full details of options available, see this link: <http://www.makelinux.com/man/1/R/raxmlHPC>

It will take ~4 hours to complete. You can check the status of your job using qstat command

```
qstat -u username
```



```
raxmlHPC-PTHREADS-SSE3  Multi-threaded RAxML version
  -T                      Number of threads to use
  -p                      Random number seed for the parsimony inferences
  -s                      Input alignment
  -n                      Output base name
  -c                      Rate categories
  -m PROT GAMMA I JTT F  Protein + Gamma + invariant sites + JTT as model + use
                          empirical AA frequency.
  -x                      Random seed to turn on rapid bootstrapping
  -N                      Number of bootstraps
  -f a                    Rapid Bootstrap analysis + search for best-scoring ML tree
                          in one program
```

Once completed you will see the following files in the directory:

```
RAxML.e1234567          # standard error, number=job id
RAxML.o1234567          # standard output
RAxML_bootstrap.results.tree  # bootstrap tree (1000 trees)
RAxML_info.results.tree    # log file
RAxML_bestTree.results.tree # Best-scoring ML tree
RAxML_bipartitions.results.tree # same but with support values
RAxML_bipartitionsBranchLabels.results.tree # same but with branch labels
```

We will just use the pre-computed results for today's workshop.

BAYESIAN TREES

For Bayesian analyses, we will use MrBayes (Altekar et al. 2004) program. For this we need nexus formatted alignment file. We also need to write MrBayes coding block to perform specific kind of analyses. Based on the ProtTest estimation, we will use Jones model with these options I+G+ F.

For doing this we need to add the following coding block to the nexus alignment file:

```
Begin mrbayes;          # start MrBayes code
log start filename=test_log.txt;  # save log file
set autoclose=yes;     # close upon completion
prset aamodelpr=fixed(jones);  # use JTT model
lset rates=invgamma;   # use G + I
prset statefreqpr=fixed(empirical);  # use F
mcmc ngen=100000 samplefreq=100 printfreq=100 nchains=4 savebrlens=yes;  #run
for 100K generations, sampling and printing every 100 gen, use 4 chains
sump burnin=12500;     # sampled parameter values
sumt burnin=12500;     # summary statistics
log stop;              # stop log
END;                   # end the code
```

Copy the mbblock file from the scripts folder and append it to the nexus alignment file.

```
cd ../MrBayes
```

```
cp ../jobfiles/mbblock ./          # copy mbblock file
```

```
cat FcC_smatrix.nex mblock >> FcC_smatrix2.nex # append block
```

Once this is done, we can run MrBayes.

```
cp ../jobfiles/step_4.2_mrbayes.sub ./ # copy submission file
```

```
cat step_4.2_mrbayes.sub # view contents
```

```
#!/bin/bash # bash script header
#PBS -q bioinformatics # bioinformatics queue
#PBS -l walltime=48:00:00 # allot 48 hrs for this job
#PBS -l nodes=1:ppn=8 # 1 node and 8 processors/node
#PBS -N MrBayes # job name
cd $PBS_O_WORKDIR # run program from present dir
module use /apps/group/bioinformatics/modules # use modules
module load mrbayes # load MrBayes program
mpirun -n 8 mb FcC_smatrix2.nex # run MrBayes
```

```
qsub step_4.2_mrbayes.sub # submit job
```

It will take ~4 hours to complete. So we will just use the pre-computed results. You can check the status of your job using qstat command

```
qstat -u username
```

Once completed you will see the following files in the directory:

```
FcC_smatrix2.nex.ckp # checkpoint file
FcC_smatrix2.nex.mcmc # statistics about MCMC run
FcC_smatrix2.nex.run1.t # sampled trees statistcs
FcC_smatrix2.nex.run2.t } mcmc # sampled trees statistcs
FcC_smatrix2.nex.run1.p # sampled parameter values
FcC_smatrix2.nex.run2.p # sampled parameter values
FcC_smatrix2.nex.pstat # parameter statistics
FcC_smatrix2.nex.lstat } sump # liklihood estimates
FcC_smatrix2.nex.con.tre # consensus trees
FcC_smatrix2.nex.parts # taxon bipartiions
FcC_smatrix2.nex.trprobs } sumt # sampled trees and probabilities
FcC_smatrix2.nex.tstat # tree statistics
FcC_smatrix2.nex.vstat # branch length statistics
MrBayes.e1234567 # standard error
MrBayes.o1234567 # standard out
test_log.txt # standard out
```

We will just use the pre-computed results for today's workshop.

VIEWING TREES

FigTree software is a graphical viewer of phylogenetic trees and can be used to produce publication-ready figures. It comes with great number of features and is platform independent (can be run on Windows, Mac or Linux machines).

You can download the executable for FigTree from here:

<http://tree.bio.ed.ac.uk/software/figtree/>

Then you can transfer results (trees) from the cluster to your local computer. Open WinSCP program (Click on start/windows icon, search for WinSCP and click on the result). Login, by selecting SCP as File protocol, coates.rcac.purdue.edu as hostname, your Purdue account **username** and **password** and clicking on Login button. Navigate to the scratch space and copy following files to your computer (/scratch/lustreA/u/**username**/phylogenomics # substitute your username and first letter of your username)

```
FcC_smatrix2.nex.con.tre           # from MrBayes direcotry
RAxML_bootstrap.results.tree       # from RAxML directory
RAxML_bestTree.results.tree        # from RAxML directory
RAxML_bipartitions.results.tree    # from RAxML directory
RAxML_bipartitionsBranchLabels.results.tree # from RAxML directory
```

Open FigTree v1.4.0 executable, which you downloaded earlier and browse to open these files separately.

REFERENCES:

- Altekar G, Dwarkadas S, Huelsenbeck JP, and Ronquist F. 2004. Parallel Metropolis coupled Markov chain Monte Carlo for Bayesian phylogenetic inference. *Bioinformatics* 20:407-415.
- Altschul SF, Gish W, Miller W, Myers EW, and Lipman DJ. 1990. Basic local alignment search tool. *J Mol Biol* 215:403-410.
- Capella-Gutierrez S, Silla-Martinez JM, and Gabaldon T. 2009. trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. *Bioinformatics* 25:1972-1973.
- Darriba D, Taboada GL, Doallo R, and Posada D. 2011. ProtTest 3: fast selection of best-fit models of protein evolution. *Bioinformatics* 27:1164-1165.
- Do CB, and Katoh K. 2008. Protein multiple sequence alignment. *Methods Mol Biol* 484:379-413.
- Dongen Sv. 2000. A cluster algorithm for graphs. Technical Report INS-R0011: National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam.
- Edgar RC. 2004. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res* 32:1792-1797.
- Fischer S, Brunk BP, Chen F, Gao X, Harb OS, Iodice JB, Shanmugam D, Roos DS, and Stoeckert CJ, Jr. 2011. Using OrthoMCL to assign proteins to OrthoMCL-DB groups or to cluster proteomes into new ortholog groups. *Curr Protoc Bioinformatics* Chapter 6:Unit 6 12 11-19.
- Kuck P, and Meusemann K. 2010. FASconCAT: Convenient handling of data matrices. *Mol Phylogenet Evol* 56:1115-1118.
- Stamatakis A. 2006. RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics* 22:2688-2690.
- Yang Y, and Smith SA. 2013. Optimizing de novo assembly of short-read RNA-seq data for phylogenomics. *BMC Genomics* 14:328.