**Divergence time estimation using MCMCTree with multiple loci.**

1. Preprocess your gene families

You will need alignments of all the gene (protein) families you are using (trimmed if desired), a best fitting substitution model, and a gene tree with branch lengths for each.

I would use IQTree because of its fancy model selection and the subsequent bits have been set up for IQTree

1. Partition your genes into families based on rate.

First, run extract\_tree\_lengths.py – you will need python modules glob and ete3 installed. You will also need to modify this line – this isn’t ideal and I’ll probably update it soon but I’m a busy boy.

treefiles = glob.glob("Pinniped\_phylogeny\_files/gene\_trees/trees/\*.treefile")

to point to where your treefiles are (as if listing them using bash.

Run using

python3 extract\_tree\_lengths.py

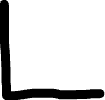
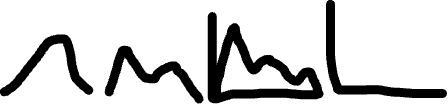
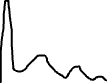
And it should make a file called tree\_lengths.csv.

Included in this directory is an R script (cluster\_fams.R) that I suggest you run line by line in Rstudio so that you can look modify it on the fly.

First you need to modify the working directory via the line

setwd("~/OneDrive - The University of Nottingham/seals/clock")

It then takes your gene trees and draws a histogram of their heights (the maximum branch length between two tips. Have a look and decide on the number of partitions you think fit the data. For example, in the professional figure below, you would decide 0, 1, 2 and 3 breaks from left to right respectively.



Modify this line with the number of clusters you have identified (it’s set at 4 now)

breaks = getJenksBreaks(dists$dist, 4, subset = NULL)

It will then draw boundaries between four populations of tree heights, which I use as partitions in mcmctree. (note – currently the lines are hardcoded with some old values and I might have forgotten to change this, so you might need to change these yourself otherwise the figure won’t make sense (the program will still work as intended – it’s just for your understanding))

It will then write “clusters.csv”, which has the gene family name and the height and the cluster.

Finally, write your partitioned alignment file in phyllip format. Run

python3 generate\_alignment.py <alignments\_directory>

, replacing the <alignments directory> with your own directory containing all the alignments with the extention “.faa”. This makes a file “partitioned\_alignment.phy” which is what you need.

1. Prepare your calibrated species tree

You need to use as many calibrations as possible – which is outside the scope of this manual (thank God). It’s down to you what to use and how to implement them, but I usually impose a uniform distribution with a hard minimum bound youngest possible age of the fossil and the a soft boundary at the maximum bound, which is usually the age at which we would expect to find evidence for the clade if it existed, but that we don’t.

I would modify the newick string for the species tree in a text editor. First remove the branch lengths and any clade labels. Next, you need a line that has the number of species at the start (not sure why – ask Ziheng), then for each calibrated node you can add a string in the following format.

'B(<X>,<Y>,1e-300,0.025)'

Where <X> is the minimum age and <Y> the soft maximum (1e-300 is the probability of the divergence time being lower than the minimum, so basically 0, and 0.025 is the probability of it being older than the maximum). I find it easiest to keep copying the newick string to FigTree and showing ranch labels to ensure I’ve been calibrating the correct nodes and that I haven’t accidentally messed with the toplogy). Your calibrated tree will look something like this.

21

(((((((bearded-seal,(hooded-seal,((largha-seal,harbour-seal),(ringed-seal,(grey-seal,caspian-seal)))))'B(1.32,2.729,1e-300,0.025)',((weddell-seal,(southern-elephant-seal,northern-elephant-seal))'B(0.505,2.729,1e-300,0.025)',hawaiian-monk-seal)'B(0.71,2.729,1e-300,0.025)')'B(1.382,2.729,1e-300,0.025)',(walrus,(northern-fur-seal,(stellar-sea-lion,californian-seal-lion))'B(0.5333,2.729,1e-300,0.025)')'B(1.597,2.729,1e-300,0.025)')'B(2.045,2.729,1e-300,0.025)',(sea-otter,sable)'B(1.957,4.88,1e-300,0.025)')'B(2.48,4.88,1e-300,0.025)',(polar-bear,giant-panda)'B(1.25,4.88,1e-300,0.025)')'B(3.38,4.88,1e-300,0.025)',dog)'B(3.771,6.609,1e-300,0.025)',cat)'B(3.771,6.609,1e-300,0.025)';

1. Run mcmctree for proteins.

Running mcmctree for protein data is complicated because of the approximate likelihood estimation it employs. You basically have to run it with the simplest possible model for each gene family first, then generate a hessian matrix for each, using the best fitting model then run it properly. For more information and an alternative guide, see [this handy github vignette by Sandra Álvarez Carretero](https://github.com/sabifo4/Tutorial_MCMCtree). Thankfully, I’ve got a bash script that runs all the different steps sequentially.

In your current directory, add the models directory and a directory within which to run the analysis.

In this latter directory set up 6 directories called clock\_1 through clock\_6.

You can also modify mcmctree.ctl – the control file - and add it to each of these 6 directories. I’ll go though each line you need to think about in the control file, whose contents follows (everything after “\*” is a comment).

seqfile = ../../partitioned\_alignment.phy \*alignment file

treefile = ../../calibrated\_tree.nwk \*topology tree file

mcmcfile = mcmc.mcmc \*mcmc output

outfile = mcmc.out \*stdout output

ndata = 4 \*number of partitions in alignment file

seqtype = 2 \*0-2 nucleotides, codons, amino acids

usedata = 2 in.BV \*0-3 no data, seq, approximate in.bv

clock = 3 \*1-3 strict clock, uncorrelated rates, autocorrelated rates

model = 0 \*0-10 JC69, K80, F81, F84, HKY85, T92, TN93, REV, UNREST, REVu, UNRESTu

alpha = 0.5 \*alpha value

ncatG = 5 \*number discrete gamma categories

cleandata = 0 \*0,1 leave ambiguous sites, remove them

BDparas = 1 1 0.1 \* Birth, Death, Sampling

kappa\_gamma = 1 1 \* alpha, beta of kappa prior gamma - irrelevant here

alpha\_gamma = 1 1 \* alpha, beta of alpha prior gamma

rgene\_gamma = 2 39.513197 1 \* total tree length / root age (mean) scaled so that alpha = 2

sigma2\_gamma = 1 1 \* alpha, beta branch rate variation prior

finetune = 1: .1 .1 .1 .1 .1 .1 \*magical trickery

print = 1 \*0-2 no mcmc sample, everything except branch lengths, everything

burnin = 200000 \*n samples discarded

sampfreq = 20000 \*how often to sample chain

nsample = 20000 \*number of samples before finish

So the first four lines are your input files and output files – you need to make sure they are pointing to the right place, recalling that the mcmctree.ctl file will be in the clock\_x directories).

ndata needs to equal the number of partitions you chose. seqtype is 2 for amino acids. usedata is how the program treats the alignment, which will change automatically where appropriate.

Clock = 2 for uncorrelated rates. Clock = 3 for autocorrelated – this is a decision you have to make - a lot of people do both and share the results for both, hoping there isn’t much difference. [There is a complex way of selecting a model too](https://dosreislab.github.io/2018/08/31/bppr-stepstones.html).

Model needs to start as 0. ncatG is the number of gamma cateogies, which governs how sites can vary in rate within a partition, cleandata is about how you treat ambiguous data. These can all stay the same unless you have a reason to change them

BDparas is the birth and death parameters and kappa\_gamma is about the transition-tranversion ratio. alpha\_gamma is how rate can vary between the clusters. Don’t change these.

rgene\_gamma is the expectation of rate and is important. The gamma refers to a distribution with 2 paramentes, alpha and beta, which is what you specify. Keep an alpha as 2, so the distribution is fairly defuse. Beta needs to be specified so that the mean of the distribution is the expected substation rate. You have to do some maths!

Mean\_rate = tree\_height / root\_age

Tree\_height I specify as the average tree height of a gene tree in the analysis.

Root\_age is the mean of the prior you put on the root (from the fossil calibrations).

Beta = 2/mean\_rate = what you should put in the second field of this line.

Leave sigma2\_gamma, finetune and print

Burnin, samplefreq, and n\_sample refer to how long to run the Markov chain for. The burnin is the number of samples before saving the values. Samplefreq is how often to take results. N\_sample is how many samples to take. Unfortunately, there is no way of knowing what these should be until the run is finished, but when the analyses have finished, you can load all the .mcmc files in tracer. If the effective sample sizes are too low and/or the dates don’t all match, you will need to run the analysis again with a larger samplefreq. N\_sample should always be 10,000 or 20,000 – you shouldn’t neeed any more to draw representative distributions.

When you’ve changed the file, put a copy in each clock\_x directory.

You also need to make a file with the best model for each partition in the order the appear in your partitioned alignment file, which you can get using IQTree with modelfinderonly, on the alignment for each partition. The only likely models in the directory supplied are JTT, WAG, LG, Dayhoff, and Blosum62, so you can just which of these does best (regardless of any +G +I etc.) in the file. Call it best\_models.txt.

Also included is mcmctree\_proteins.sh, which is a slurm script that does everything you need to do. You will need to modify it to work with your job scheduler if it is not slurm. And also change this line:

for i in {0001..XXXX}

Change the XXXX to the number of gene families with leading “0”s. and

cd /your/favourite/directory/clock\_${SLURM\_ARRAY\_TASK\_ID}

with your directory. Put this in the same directory that your clock\_1 through clock\_6 are contained in and submit the job.

If you have any issues, just make sure the scripts are pointing to files that exist – you might have to add or remove a ../ or something. You can also ask me, but have a go at it first.

Done – turns out that’s a lot more long-winded than expected.