

R script to perform all analyses as presented in The megaherbivore gap after the non-avian dinosaur extinctions modified trait evolution and diversification of tropical palms

by **Renske Onstein**, email: onsteinre@gmail.com for questions

Data:

- Tree file: phylogenetic maximum clade credibility tree of palms.
- Trait data file including fruit length and armature data for extant palm species.
- Fossil data file including maximum fruit and seed size data for palm fossils in different epochs

R Packages:

Geiger, ape, phytools, diversitree, reshape2

First - getting ready!

Clear your R workspace

```
rm(list=ls())
```

Install and load the relevant R packages

```
library(geiger)
library(ape)
library(phytools)
library(diversitree)
library(reshape2)
```

Set your working directory to the folder which contains the data. Use Session -> Set working directory -> choose directory -> find the folder on your computer. For example:

```
setwd("~/Documents/Onstein")
```

To find the path to a particular file, you can use 'file.choose()'

Now, let's import the phylogenetic data:

```
tree_full<-read.nexus("TREE") # this is the MCC phylogenetic tree
tree_full<-ladderize(tree_full)
trees<-read.nexus("TREES.nex") # dit is de posterior distribution of phylogenetic trees
```

Import trait data:

```
traits<-read.csv("trait_data_Onstein_et_al.csv")
traits$log_AverageFruitLength_cm<-log(traits$AverageFruitLength_cm) #log transform fruit length data
rownames(traits)<-traits[,1]
```

Import fossil data:

```
fossils<-read.csv("fossils_size_type_v2.csv")
fossils$Midpointage2<-(fossils$Midpointage-105.39)*-1 # adjust to reversed palm time line based on age
fruit<-subset(fossils, fossils$Type=="fruit")
seed<-subset(fossils, fossils$Type=="seed")
```

1. H1 - fruit length trait space

Select trait of interest (e.g., column 7 for 'log-transformed fruit size')

```
log_fruit_size<-traits[, c(1,7)]
log_fruit_size<-log_fruit_size[complete.cases(log_fruit_size),] #remove species without fruit length data
data.frame(log_fruit_size[, 2])> dd
row.names(dd)<- log_fruit_size[,1]
```

Match names of tips in phylogenetic data with data names in trait file. Drop these species from the tree or data, because we need matching data between tree and traits:

```
x<-name.check(tree_full, dd)
drop<-drop.tip(tree_full, x$tree_not_data)
log_fruit_size2<-log_fruit_size[!log_fruit_size$SpecName %in% x$data_not_tree, ]
```

Check again whether data matches now

```
data.frame(log_fruit_size2[, 2])> dd2
row.names(dd2)<- log_fruit_size2[,1]
dd2
x<-name.check(drop, dd2) #OK
```

Prepare data for models

```
x<-log_fruit_size2[,2]
names(x)<-rownames(log_fruit_size2)
tree<-drop
```

Run the fastAnc function to estimate ancestral states

```
aa <- fastAnc(tree, x, CI=TRUE) # run ancestral state reconstruction
aa$ace # have a look at reconstructed state at internal nodes - root state = 1.188
```

Run the fancyTree function to estimate the trait space and confidence intervals, plot the fossil data in there, and save the plot as a pdf

```
pdf("traitgram_fancy_log.pdf")
par(mfrow=c(1,1))
fancyTree(tree, type = "phenogram95", x = x, fsize = 0.8)
abline(v=c(39.9,65.39),lty="dashed")
points(log(fruit$Fossil_size)~fruit$Midpointage2, data=fruit, pch=15, col="red", cex=0.8)
points(log(seed$Fossil_size)~seed$Midpointage2, data=seed, pch=17, col="red", cex=0.8)
dev.off()
```

Including *Nypa* fossil constraint. We will now add an extant tip ("*Nipadites_sp*") sister to taxon '*Nypa fruticans*' with divergence time of 52.5 Ma (the age of the fossil)

```
node2 <- which(tree$tip.label=="Nypa_fruticans")
tree2 <- bind.tip(tree, tip.label="Nipadites_sp", where=node2,
                  position=52.5, edge.length=0.1)
log_fruit_size3 <- rbind(log_fruit_size2, "Nipadites_sp" = c("Nipadites_sp", as.numeric(log(17.78))))
log_fruit_size3 $ log_AverageFruitLength_cm<-as.numeric(log_fruit_size3 $ log_AverageFruitLength_cm)
```

Prepare data and run the ancestral state reconstruction model:

```
x<-log_fruit_size3[,2]
names(x)<-rownames(log_fruit_size3)
aa <- fastAnc(tree2, x, CI=TRUE) # run ancestral state reconstruction
aa$ace # have a look at reconstructed state at internal nodes - root state = 1.404
```

Run the fancyTree function to estimate the trait space and confidence intervals, plot the fossil data in there, and save the plot as a pdf

```
pdf("traitgram_fancy_fossil_inclNypa_log.pdf")
fancyTree(tree2, type = "phenogram95", x = x, fsize = 0.8)
abline(v=c(39.9,65.39),lty="dashed")
points(log(fruit$Fossil_size)~fruit$Midpointage2, data=fruit, pch=15, col="red", cex=0.8)
points(log(seed$Fossil_size)~seed$Midpointage2, data=seed, pch=17, col="red", cex=0.8)
dev.off()
```

2. H1 - armature ancestral state reconstruction

Select trait of interest (e.g., column 3 for '*armature*')

```
armature<-traits[, c(1,3)]
armature<-armature[complete.cases(armature),]
data.frame(armature[, 2])> dd
row.names(dd)<- armature[,1]
```

Prepare data - drop tips if needed

```
x<-name.check(tree_full, dd)
drop<-drop.tip(tree_full, x$tree_not_data)
armature2<-armature[!armature$SpecName %in% x$data_not_tree, ]
data.frame(armature2[, 2])> dd2
row.names(dd2)<- armature2[,1]
dd2
```

```
x<-name.check(drop, dd2)
tree<-ladderize(drop, right = FALSE)
x<-armature2[,2]
names(x)<-rownames(armature2)
```

Simulating stochastic character histories for a discrete character (stochastic character mapping) We sample character histories from their posterior probability distribution.

```
mtree <- make.simap(tree, x, model = "ARD")
cols <- setNames(c("grey", "gold"), sort(unique(x)))
plotSimmap(mtree, cols, pts = FALSE, lwd = 2)
add.simap.legend(colors = cols, vertical = FALSE, prompt = FALSE, x = 0, y = 24)
axisPhylo()
```

A single stochastic character map does not mean a whole lot in isolation, we therefore simulate n=500 times

```
mtrees <- make.simap(tree, x, model="ARD", nsim = 500)
XX2 <- describe.simap(mtrees, plot = FALSE)
cols2 <- setNames(c("grey", "gold"), sort(unique(x)))
nodelabels(pie = XX2$ace, piecol = cols2, cex = 0.3)
```

We can also use stochastic mapping to plot the posterior probability that the edges & nodes of the tree are in a binary state. We first have to convert our mapped edge states to 0 & 1.

```
stateb <- mergeMappedStates(mtrees, "0", "0")
stateb <- mergeMappedStates(stateb, "1", "1")
```

Now plot a density map

```
XX <- densityMap(stateb, lwd = 2)
obj<-XX
```

Change colour scheme if needed and save as pdf

```
n<-length(obj$cols) ## what is the length of the current color ramp?
obj$cols[1:n]<-colorRampPalette(c("grey","black", "gold"), space="Lab")(n)
pdf("ASR_armature_plot.pdf")
plot(obj, lwd=2)
axisPhylo()
dev.off()
```

3. H2 and H3 - diversification rate models for fruits

Select trait of interest (e.g., column 6 for 'binary fruit size')

```
fruit_binary<-traits[, c(1,6)]
fruit_binary<-fruit_binary[complete.cases(fruit_binary),]
data.frame(fruit_binary[, 2])> dd
row.names(dd)<- fruit_binary[,1]
```

Prepare data

```
x<-name.check(tree_full, dd)
drop<-drop.tip(tree_full, x$tree_not_data)
fruit_binary2<-fruit_binary[!fruit_binary$SpecName %in% x$data_not_tree, ]
```

Drop tips from set of trees as well

```
allreps <- list()
for(i in 1:100)
{
  thisrep <- drop.tip(trees[[i]],x$tree_not_data)
  allreps[[i]] <- thisrep
}

trees2<-allreps
```

Data and tree should match now:

```
data.frame(fruit_binary2[, 2])> dd2
row.names(dd2)<- fruit_binary2[,1]
dd2
x<-name.check(drop, dd2)
```

Prepare data for running the models:

```
tree<-ladderize(drop, right = TRUE)
x<-fruit_binary2[,2]
names(x)<-rownames(fruit_binary2)
fruit.v <- x
tree3 <- tree
```

Model selection

Simple BiSSE model:

```
lik <- make.bisse(tree3, fruit.v)
p <- starting.point.bisse(tree3)
fit <- find.mle(lik, p)
fit$lnLik
round(coef(fit), 3)
```

Bisse time-dependent model:

Define time slices

```
TP1<-40 # 40 Ma
TP2<-66 # K-Pg events
```

Model:

```
lik.t <- make.bisse.td(tree3,fruit.v, 3)
argnames(lik.t)
pars.t <- c(40, 66, fit$par, fit$par, fit$par)
names(pars.t) <- argnames(lik.t)
```

Fit full model

```
lik.t2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2)
fit.t <- find.mle(lik.t2, pars.t[argnames(lik.t2)],control=list(maxit=20000))
```

Are rates constant from small/large fruits in period 1 and 3? → We hypothesize that because period 1 and 3 share megaherbivores, rates should be the same. We therefore constraint period 1 and 3 in terms of speciation, extinction and transition rates of large vs. small fruits

Constrain lambdas, mus and qs for period 1 and 3, for large vs small fruits

```
lik.t13 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3, mu0.1 ~ mu0.3, mu1.1 ~ mu1.3)
fit.t13 <- find.mle(lik.t13, pars.t[argnames(lik.t13)],control=list(maxit=20000))
anova(fit.t, equal.t13=fit.t13) # supported
```

Are rates (speciation/extinction) constant for small/large fruits in period 2? → We hypothesize that rates of small and large fruits could be similar in period 2, due to absence of megaherbivores.

In addition, is speciation and extinction in period 2 the same for small/large fruits?

```
lik.t13_lm2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3)
fit.t13_lm2 <- find.mle(lik.t13_lm2, pars.t[argnames(lik.t13_lm2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_lm2=fit.t13_lm2) # supported
```

or just speciation? (if previous model - t13_lm2 - not supported)

```
lik.t13_l2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3)
fit.t13_l2 <- find.mle(lik.t13_l2, pars.t[argnames(lik.t13_l2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_l2=fit.t13_l2)
```

or just extinction? (if previous model - t13_lm2 - not supported)

```
lik.t13_m2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3)
fit.t13_m2 <- find.mle(lik.t13_m2, pars.t[argnames(lik.t13_m2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_m2=fit.t13_m2)
```

Are ALL rates constant in period 1 and 3 (for small/large fruits)? → We hypothesize that rates of small and large fruits differ in period 1 and 3,

In addition, constrain lambdas, mus and qs for large vs small fruits in period 1/3

```
lik.t13_2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda0.3, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2)
fit.t13_2 <- find.mle(lik.t13_2, pars.t[argnames(lik.t13_2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_2=fit.t13_2) # not supported
```

or just lambdas?

```
lik.t13_l <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ 1.
fit.t13_l <- find.mle(lik.t13_l, pars.t[argnames(lik.t13_l)],control=list(maxit=20000))
anova(fit.t13_lm2, equal.t13_l=fit.t13_l) # not supported
```

or just mus?

```
lik.t13_m <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ 1.
fit.t13_m <- find.mle(lik.t13_m, pars.t[argnames(lik.t13_m)],control=list(maxit=20000))
anova(fit.t13_lm2, equal.t13_m=fit.t13_m) # supported
```

or just qs?

```
lik.t13_q <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ 1.
fit.t13_q <- find.mle(lik.t13_q, pars.t[argnames(lik.t13_q)],control=list(maxit=20000))
anova(fit.t13_m2, equal.t13_q=fit.t13_q) # not supported
```

Are transition rates (for small/large fruits) different in period 2? -> We hypothesize that transition rates differ in period 2, due to changing selection pressures,

Now, can we also constrain q in period 2? or are they indeed different, as predicted?

```
lik.t14 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q01.2 ~ q10.2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, 1.
fit.t14 <- find.mle(lik.t14, pars.t[argnames(lik.t14)],control=list(maxit=20000))
anova(fit.t13_m, equal.t14=fit.t14) # not supported
```

Final additional constraints supported? what about speciation, extinction and transitions in period 2 to be similar to rates in period 1/3?

speciation fruits period 2 to small fruits period 1/3

```
lik.t15_10 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2~lambda0.3, lambda1.2 ~ lambda0.3, mu0.2
fit.t15_10 <- find.mle(lik.t15_10, pars.t[argnames(lik.t15_10)],control=list(maxit=20000))
anova(fit.t13_m, equal.t15_10=fit.t15_10) # not supported
```

speciation fruits period 2 to large fruits period 1/3

```
lik.t15_l1 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2~lambda1.3, lambda1.2 ~ lambda1.3, mu0.2
fit.t15_l1 <- find.mle(lik.t15_l1, pars.t[argnames(lik.t15_l1)],control=list(maxit=20000))
anova(fit.t13_m, equal.t15_l1=fit.t15_l1) # supported
```

extinction fruits period 2 to fruits period 1/3

```
lik.t15_m <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2~mu0.3, mu1.2 ~ mu0.3, lambda0.2 ~ lambda1.2,
fit.t15_m <- find.mle(lik.t15_m, pars.t[argnames(lik.t15_m)],control=list(maxit=20000))
anova(fit.t13_m, equal.t15_m=fit.t15_m) # supported
```

all transition fruits period 2 similar to rates period 1/3

```
lik.t15_q <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q01.2~q01.3, q10.2~q10.3, lambda0.2 ~ lambda1.2, mu0
fit.t15_q <- find.mle(lik.t15_q, pars.t[argnames(lik.t15_q)],control=list(maxit=20000))
anova(fit.t13_m, equal.t15_q=fit.t15_q) # not supported
```

only forward transition 1

```
lik.t15_q01 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q01.2~q01.3, lambda1.2 ~ lambda1.3, mu0.2 ~ mu1.2,
fit.t15_q01 <- find.mle(lik.t15_q01, pars.t[argnames(lik.t15_q01)], control=list(maxit=20000))
anova(fit.t13_m, equal.t15_q01=fit.t15_q01) # not supported
```

only forward transition 2

```
lik.t15_q012 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q01.2~q10.3, lambda1.2 ~ lambda1.3, mu0.2 ~ mu1.2,
fit.t15_q012 <- find.mle(lik.t15_q012, pars.t[argnames(lik.t15_q012)], control=list(maxit=20000))
anova(fit.t13_m, equal.t15_q012=fit.t15_q012) # not supported
```

only backward transition 1

```
lik.t15_q10 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3, lambda1.2 ~ lambda1.3, mu0.2 ~ mu1.2,
fit.t15_q10 <- find.mle(lik.t15_q10, pars.t[argnames(lik.t15_q10)], control=list(maxit=20000))
anova(fit.t13_m, equal.t15_q10=fit.t15_q10) # supported
```

only backward transition 2

```
lik.t15_q102 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q01.3, lambda1.2 ~ lambda1.3, mu0.2 ~ mu1.2,
fit.t15_q102 <- find.mle(lik.t15_q102, pars.t[argnames(lik.t15_q102)], control=list(maxit=20000))
anova(fit.t13_m, equal.t15_q102=fit.t15_q102) # supported
```

A combination of the last supported models:

speciation and extinction fruits period 2 to large fruits period 1/3 and backward transition 1

```
lik.t15_l1q10 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3, lambda0.2~lambda1.3, lambda1.2 ~
fit.t15_l1q10 <- find.mle(lik.t15_l1q10, pars.t[argnames(lik.t15_l1q10)], control=list(maxit=20000))
anova(fit.t13_m, equal.t15_l1q10=fit.t15_l1q10) # supported
```

speciation and extinction fruits period 2 to large fruits period 1/3 and backward transition 2

```
lik.t15_l1q102 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q01.3, lambda0.2~lambda1.3, lambda1.2 ~
fit.t15_l1q102 <- find.mle(lik.t15_l1q102, pars.t[argnames(lik.t15_l1q102)], control=list(maxit=20000))
anova(fit.t13_m, equal.t15_l1q102=fit.t15_l1q102) # supported
```

Comparison of fit for all models tested

```
x<-anova(fit.t, equal.t13=fit.t13, equal.t13_lm2=fit.t13_lm2, equal.t13_2=fit.t13_2, equal.t13_1=fit.t13_1)
```

Save models to file and save workspace

```
write.csv(x, file="models_fruits.csv")
save.image("Workspace Model Testing Fruits.RData")
```

Run the best model in a Bayesian framework

Prepare starting parameters


```
lik.t <- make.bisse.td(trees2[[1]],fruit.v, 3)
argnames(lik.t)
pars.t <- c(40, 66, fit$par, fit$par, fit$par)
names(pars.t) <- argnames(lik.t)
```

Best model fruits: lik.t15_l1q10 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3, lambda0.2~lambda1.3, lambda1.2 ~ lambda1.3, mu0.2~mu0.3, mu1.2 ~ mu0.3, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3, mu0.1 ~ mu0.3, mu1.1 ~ mu0.3, mu1.3 ~ mu0.3, q01.1 ~ q01.3, q10.1 ~ q10.3)

```
lik <- make.bisse(trees2[[1]], fruit.v)
p <- starting.point.bisse(trees[[1]])
fit <- find.mle(lik, p)
```

get ready - optimize parameter settings

```
funcb <- make.bisse.td(trees2[[1]],fruit.v, 3)
func <- constrain(funcb, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3, lambda0.2~lambda1.3, lambda1.2 ~ lambda1.3
start2 <- starting.point.bisse(tree=trees2[[1]])
start <- c(40, 66, fit$par, fit$par, fit$par)
names(start) <- argnames(funcb)
prior <- make.prior.exponential(2*(log(2500)/116)) # number of tips given total age of phylogeny
```

optimize w (short run)

```
tmp4w <- mcmc(func, start[argnames(func)], prior=prior, nsteps=100, w=0.1)
w <- diff(sapply(tmp4w[2:7], quantile, c(.05, .95))); w
```

Loop through 100 phylogenetic trees

```
mcmc.PALMSfruit.time <- list()

for(i in seq(from=1, to=100, by=1)){
  print(i)
  tree <- trees2[[i]]
  funcb <- make.bisse.td(trees2[[i]],fruit.v, 3)
  func <- constrain(funcb, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3, lambda0.2~lambda1.3, lambda1.2 ~ lambda1.3
  start2 <- starting.point.bisse(tree=trees[[i]])
  start <- c(40, 66, fit$par, fit$par, fit$par)
  names(start) <- argnames(funcb)
  prior <- make.prior.exponential(2*(log(2500)/116))
  thisRep <- mcmc(func, start[argnames(func)], prior=prior, nsteps=100, w=w)
  mcmc.PALMSfruit.time[[i]] <- thisRep
  save.image("fruit_model_t15_l1q10.RData")
}
```

Extract data

```
newlist<-(mcmc.PALMSfruit.time[1:100])

unconstr <- list()
for (i in seq(1,100,1)){
```

```

  unconstr[[i]] <- newlist[[i]]
}

unconstr1b<-Filter(function(x) {nrow(x) >= 100}, unconstr)
length(unconstr1b)

summ_uc <- matrix(ncol=8, nrow=length(unconstr)*length(10:100))
colnames(summ_uc) <- names(unconstr[[1]])

for (par in 1:8){
  allRep<- NULL
  for (rep in 1:length(unconstr)){
    thisRep <- unconstr[[rep]][par][10:100,]
    allRep <- append(allRep, thisRep)
  }
  summ_uc[,par] <- allRep
}; rm("par", "rep", "thisRep", "allRep")
summ_uc <- data.frame(summ_uc)

```

Calculate speciation, extinction, transition and net diversification rates given constraints in model

```

summ_uc$lambda0.1<-summ_uc$lambda0.3
summ_uc$lambda1.1<-summ_uc$lambda1.3
summ_uc$lambda0.1<-summ_uc$lambda0.3
summ_uc$lambda0.2<-summ_uc$lambda1.3
summ_uc$lambda1.2<-summ_uc$lambda1.3
summ_uc$mu0.1<-summ_uc$mu0.3
summ_uc$mu1.1<-summ_uc$mu0.3
summ_uc$mu0.2<-summ_uc$mu0.3
summ_uc$mu1.2<-summ_uc$mu0.3
summ_uc$mu1.3<-summ_uc$mu0.3
summ_uc$r0.1=summ_uc$lambda0.1-summ_uc$mu0.1
summ_uc$r0.2=summ_uc$lambda0.2-summ_uc$mu0.2
summ_uc$r0.3=summ_uc$lambda0.3-summ_uc$mu0.3
summ_uc$r1.1=summ_uc$lambda1.1-summ_uc$mu1.1
summ_uc$r1.2=summ_uc$lambda1.2-summ_uc$mu1.2
summ_uc$r1.3=summ_uc$lambda1.3-summ_uc$mu1.3
summ_uc$q01.1<-summ_uc$q01.3
summ_uc$q10.1<-summ_uc$q10.3
summ_uc$q10.2<-summ_uc$q10.3

```

Save dataframe

```
write.csv(summ_uc, file="mcmc.PALMSfruit.time_100trees_lik_tm_15.csv")
```

Plot densities of rates

```

par(mfrow=c(2,2))
col <- c("#004165", "#eaab00", "red", "pink", "black", "grey")
profiles.plot(summ_uc[c("lambda0.1", "lambda0.2", "lambda0.3", "lambda1.1", "lambda1.2", "lambda1.3")],
profiles.plot(summ_uc[c("q01.1", "q01.2", "q01.3", "q10.1", "q10.2", "q10.3")], col.line=col, las=1,xlab="Divergence")
profiles.plot(summ_uc[c("r0.1", "r0.2", "r0.3", "r1.1", "r1.2", "r1.3")], col.line=col, las=1,xlab="Divergence")

```

Plot boxplots - prepare data

```
combi<-summ_uc
```

```
str(combi)
combi$X<- NULL
combi$i <- NULL
combi$p<- NULL
```

```
# lambda
lambda<-combi
lambda$mu0.1<- NULL
lambda$mu0.2<- NULL
lambda$mu0.3<- NULL
lambda$mu1.1<- NULL
lambda$mu1.2<- NULL
lambda$mu1.3<- NULL
lambda$q01.1 <- NULL
lambda$q10.1 <- NULL
lambda$q01.2 <- NULL
lambda$q10.2 <- NULL
lambda$q01.3 <- NULL
lambda$q10.3 <- NULL
lambda$r0.1<-NULL
lambda$r0.2<-NULL
lambda$r0.3<-NULL
lambda$r1.1<-NULL
lambda$r1.2<-NULL
lambda$r1.3<-NULL
```

```
# mu
mu<-combi
mu$lambda0.1<- NULL
mu$lambda0.2<- NULL
mu$lambda0.3<- NULL
mu$lambda1.1<- NULL
mu$lambda1.2<- NULL
mu$lambda1.3<- NULL
mu$q01.1 <- NULL
mu$q10.1 <- NULL
mu$q01.2 <- NULL
mu$q10.2 <- NULL
mu$q01.3 <- NULL
mu$q10.3 <- NULL
mu$r0.1<-NULL
mu$r0.2<-NULL
mu$r0.3<-NULL
mu$r1.1<-NULL
mu$r1.2<-NULL
mu$r1.3<-NULL
```

```
# q
q<-combi
q$lambda0.1<- NULL
```

```

q$lambda1.1 <- NULL
q$lambda0.2<- NULL
q$lambda1.2 <- NULL
q$lambda0.3<- NULL
q$lambda1.3 <- NULL
q$mu0.1<- NULL
q$mu0.2<- NULL
q$mu0.3<- NULL
q$mu1.1<- NULL
q$mu1.2<- NULL
q$mu1.3<- NULL
q$r0.1<-NULL
q$r0.2<-NULL
q$r0.3<-NULL
q$r1.1<-NULL
q$r1.2<-NULL
q$r1.3<-NULL

```

```

# r
r<-combi
r$lambda0.1<- NULL
r$lambda1.1 <- NULL
r$lambda0.2<- NULL
r$lambda1.2 <- NULL
r$lambda0.3<- NULL
r$lambda1.3 <- NULL
r$mu0.1<- NULL
r$q01.1 <- NULL
r$q10.1 <- NULL
r$q01.2 <- NULL
r$q10.2 <- NULL
r$q01.3 <- NULL
r$q10.3 <- NULL
r$mu0.1<- NULL
r$mu0.2<- NULL
r$mu0.3<- NULL
r$mu1.1<- NULL
r$mu1.2<- NULL
r$mu1.3<- NULL

```

```

# stack data (reshape)
stacked.data1 = melt(lambda)
stacked.data2 = melt(mu)
stacked.data3 = melt(q)
stacked.data4 = melt(r)

```

Plot and save as pdf

```
par(mfrow=c(2,2))
```

```

pdf("fruit_boxplot_rates.pdf")
boxplots.triple = boxplot((value)~ variable, data = stacked.data1, cex=0.8, las=2, cex.lab=0.2)
boxplots.triple = boxplot((value)~ variable, data = stacked.data2, cex=0.8, las=2, cex.lab=0.2)
boxplots.triple = boxplot((value)~ variable, data = stacked.data3, cex=0.8, las=2, cex.lab=0.2)

```

```
boxplots.triple = boxplot((value)~ variable, data = stacked.data4, cex=0.8, las=2, cex.lab=0.2)
dev.off()
```

4. H2 and H3 - diversification rate models for armature

Select trait of interest (e.g., column 3 for 'armature')

```
armature<-traits[, c(1,3)]
armature<-armature[complete.cases(armature),]
data.frame(armature[, 2])> dd
row.names(dd)<- armature[,1]
```

Prepare data

```
x<-name.check(tree_full, dd)
drop<-drop.tip(tree_full, x$tree_not_data)
armature2<-armature[!armature$SpecName %in% x$data_not_tree, ]
```

drop tips from set of trees as well

```
allreps <- list()
for(i in 1:100)
{
  thisrep <- drop.tip(trees[[i]],x$tree_not_data)
  allreps[[i]] <- thisrep
}
```

```
trees2<-allreps
```

Now the data should match the phylogen

```
data.frame(armature2[, 2])> dd2
row.names(dd2)<- armature2[,1]
dd2
x<-name.check(drop, dd2)
```

Prepare data for models:

```
tree<-ladderize(drop, right = TRUE)
x<-armature2[,2]
names(x)<-rownames(armature2)
spines.v <- x
tree3 <- tree
```

Model selection

Simple BiSSE model

```
lik <- make.bisse(tree3, spines.v)
p <- starting.point.bisse(tree3)
fit <- find.mle(lik, p)
fit$lnLik
round(coef(fit), 3)
```

Bisse time-dependent model: Define time slices

```
TP1<-40 # 40 Ma
TP2<-66 # K-Pg events
```

Model:

```
lik.t <- make.bisse.td(tree3,spines.v, 3)
argnames(lik.t)
pars.t <- c(40, 66, fit$par, fit$par, fit$par)
names(pars.t) <- argnames(lik.t)
```

Fit full model

```
lik.t2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2)
fit.t <- find.mle(lik.t2, pars.t[argnames(lik.t2)],control=list(maxit=20000))
```

Are rates constant from armature/no armature in period 1 and 3? -> We hypothesize that because period 1 and 3 share megaherbivores, rates should be the same. We therefore constraint period 1 and 3 in terms of speciation, extinction and transition rates of armature vs. no armature

Constrain lambdas, mus and qs for period 1 and 3, for armature/no armature

```
lik.t13 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3, mu0.1 ~ mu0.3, mu1.1 ~ mu1.3, q0.1 ~ q0.3, q1.1 ~ q1.3)
fit.t13 <- find.mle(lik.t13, pars.t[argnames(lik.t13)],control=list(maxit=20000))
anova(fit.t, equal.t13=fit.t13) # supported
```

Are rates (speciation/extinction) constant for armature/no armature in period 2? -> We hypothesize that rates of armature and no armature could be similar in period 2, due to absence of megaherbivores.

In addition, is speciation and extinction in period 2 the same for armature/no armature?

```
lik.t13_lm2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3, mu0.1 ~ mu0.3, mu1.1 ~ mu1.3, q0.1 ~ q0.3, q1.1 ~ q1.3)
fit.t13_lm2 <- find.mle(lik.t13_lm2, pars.t[argnames(lik.t13_lm2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_lm2=fit.t13_lm2) # not supported
```

or just speciation? (if previous model - t13_lm2 - not supported)

```
lik.t13_l2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2 ~ lambda1.2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3, mu0.1 ~ mu0.3, mu1.1 ~ mu1.3, q0.1 ~ q0.3, q1.1 ~ q1.3)
fit.t13_l2 <- find.mle(lik.t13_l2, pars.t[argnames(lik.t13_l2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_l2=fit.t13_l2) # not supported
```

or just extinction? (if previous model - t13_lm2 - not supported)

```
lik.t13_m2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda1.3, mu0.1 ~ mu0.3, mu1.1 ~ mu1.3, q0.1 ~ q0.3, q1.1 ~ q1.3)
fit.t13_m2 <- find.mle(lik.t13_m2, pars.t[argnames(lik.t13_m2)],control=list(maxit=20000))
anova(fit.t13, equal.t13_m2=fit.t13_m2) # supported
```

Are ALL rates constant in period 1 and 3 (for armature/no armature)? -> We hypothesize that rates of armature and no armature differ in period 1 and 3,

In addition, constrain lambdas, mus and qs for armature/no armature in period 1/3

```
lik.t13_2 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ 1.
fit.t13_2 <- find.mle(lik.t13_2, pars.t[argnames(lik.t13_2)],control=list(maxit=20000))
anova(fit.t13_m2, equal.t13_2=fit.t13_2) # not supported
```

or just lambdas?

```
lik.t13_l <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ 1.
fit.t13_l <- find.mle(lik.t13_l, pars.t[argnames(lik.t13_l)],control=list(maxit=20000))
anova(fit.t13_m2, equal.t13_l=fit.t13_l) # supported
```

or just mus?

```
lik.t13_m <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ 1.
fit.t13_m <- find.mle(lik.t13_m, pars.t[argnames(lik.t13_m)],control=list(maxit=20000))
anova(fit.t13_m2, equal.t13_m=fit.t13_m) # not supported
```

or just qs?

```
lik.t13_q <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3, lambda1.1 ~ 1.
fit.t13_q <- find.mle(lik.t13_q, pars.t[argnames(lik.t13_q)],control=list(maxit=20000))
anova(fit.t13_m2, equal.t13_q=fit.t13_q) # not supported
```

Are transition rates (for armature/no armature) different in period 2? -> We hypothesize that transition rates differ in period 2, due to changing selection pressures,

Now, can we also constrain q in period 2? or are they indeed different, as predicted?

```
lik.t14 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q01.2 ~ q10.2, mu0.2 ~ mu1.2, lambda0.1 ~ lambda0.3,
fit.t14 <- find.mle(lik.t14, pars.t[argnames(lik.t14)],control=list(maxit=20000))
anova(fit.t13_l, equal.t14=fit.t14) # supported
```

Final additional constraints supported? what about speciation, extinction and transitions in period 2 to be similar to rates in period 1/3?

speciation no spines period 2

```
lik.t15_l0 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda0.2~lambda0.3, q01.2 ~ q10.2, mu0.2 ~ mu1.2,
fit.t15_l0 <- find.mle(lik.t15_l0, pars.t[argnames(lik.t15_l0)],control=list(maxit=20000))
anova(fit.t14, equal.t15_l0=fit.t15_l0) # not supported
```

speciation spines period 2

```
lik.t15_l1 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, lambda1.2~lambda0.3, q01.2 ~ q10.2, mu0.2 ~ mu1.2,
fit.t15_l1 <- find.mle(lik.t15_l1, pars.t[argnames(lik.t15_l1)],control=list(maxit=20000))
anova(fit.t14, equal.t15_l1=fit.t15_l1) # not supported
```

extinction period 2 to no spines extinction 1/3

```
lik.t15_m0 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu1.2~mu0.3, mu0.2~mu0.3, q01.2 ~ q10.2, lambda0.1
fit.t15_m0 <- find.mle(lik.t15_m0, pars.t[argnames(lik.t15_m0)],control=list(maxit=20000))
anova(fit.t14, equal.t15_m0=fit.t15_m0) # supported
```

extinction period 2 to spines extinction 1/3

```
lik.t15_m1 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, mu1.2~mu1.3, q01.2 ~ q10.2, mu0.2 ~ mu1.3, lambda0
fit.t15_m1 <- find.mle(lik.t15_m1, pars.t[argnames(lik.t15_m1)],control=list(maxit=20000))
anova(fit.t14, equal.t15_m1=fit.t15_m1) # not supported
```

transition period 2 to no spines transition 1/3

```
lik.t15_q0 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q01.3, q01.2 ~ q01.3, mu0.2 ~ mu1.2, lambda0
fit.t15_q0 <- find.mle(lik.t15_q0, pars.t[argnames(lik.t15_q0)],control=list(maxit=20000))
anova(fit.t14, equal.t15_q0=fit.t15_q0) # not supported
```

transition period 2 to spines transition 1/3

```
lik.t15_q1 <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3,q01.2 ~ q10.3, mu0.2 ~ mu1.2, lambda0.
fit.t15_q1 <- find.mle(lik.t15_q1, pars.t[argnames(lik.t15_q1)],control=list(maxit=20000))
anova(fit.t14, equal.t15_q1=fit.t15_q1) # supported
```

a combination of the last supported models:

transition period 2 to spines transition 1/3 and extinction period 2 to no spines extinction 1/3

```
lik.t15_q1m <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3,q01.2 ~ q10.3, mu1.2~mu0.3, mu0.2~mu0
fit.t15_q1m <- find.mle(lik.t15_q1m, pars.t[argnames(lik.t15_q1m)],control=list(maxit=20000))
anova(fit.t14, equal.t15_q1m=fit.t15_q1m) # supported
```

Comparison of fit for all models tested

```
x<-anova(fit.t, equal.t13=fit.t13, equal.t13_lm2=fit.t13_lm2, equal.t13_l2=fit.t13_l2, equal.t13_m2=fit
```

Save models to file and save workspace

```
write.csv(x, file="models_armature.csv")
save.image("Workspace Model Testing Armature.RData")
```

Run the best model in a Bayesian framework

Prepare starting parameters

```
lik.t <- make.bisse.td(trees2[[1]],spines.v, 3)
argnames(lik.t)
pars.t <- c(40, 66, fit$par, fit$par, fit$par)
names(pars.t) <- argnames(lik.t)
```

Best model armature: lik.t15_q1m <- constrain(lik.t, t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3,q01.2 ~ q10.3, mu1.2~mu0.3, mu0.2~mu0.3, lambda0.1 ~ lambda0.3, lambda1.1 ~ lambda0.3,lambda1.3 ~ lambda0.3, mu0.1 ~ mu0.3, mu1.1 ~ mu1.3, q01.1 ~ q01.3, q10.1 ~ q10.3)

```
lik <- make.bisse(trees2[[1]], spines.v)
p <- starting.point.bisse(trees[[1]])
fit <- find.mle(lik, p)
```


get ready - optimize parameter settings

```
funcb <- make.bisse.td(trees2[[1]],spines.v, 3)
func <- constrain(funcb,t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3,q01.2 ~ q10.3, mu1.2~mu0.3, mu0.2~mu0.3, lambda
start2 <- starting.point.bisse(tree=trees2[[1]])
start <- c(40, 66, fit$par, fit$par, fit$par)
names(start) <- argnames(funcb)
prior <- make.prior.exponential(2*(log(2500)/116)) # number of tips given total age of phylogeny
```

optimize w (short run)

```
tmp4w <- mcmc(func, start[argnames(func)], prior=prior, nsteps=100, w=0.1)
w <- diff(sapply(tmp4w[2:8], quantile, c(.05, .95))); w
```

loop through 100 phylogenetic trees

```
mcmc.PALMSSpines.time <- list()

for(i in seq(from=1, to=100, by=1)){
  print(i)
  tree <- trees2[[i]]
  funcb <- make.bisse.td(trees2[[i]],spines.v, 3)
  func <- constrain(funcb,t.1 ~ TP1, t.2 ~ TP2, q10.2~q10.3,q01.2 ~ q10.3, mu1.2~mu0.3, mu0.2~mu0.3, lambda
  start2 <- starting.point.bisse(tree=trees2[[i]])
  start <- c(40, 66, fit$par, fit$par, fit$par)
  names(start) <- argnames(funcb)
  prior <- make.prior.exponential(2*(log(2500)/116))
  thisRep <- mcmc(func, start[argnames(func)], prior=prior, nsteps=100, w=w) #220 so 10%=200
  mcmc.PALMSSpines.time[[i]] <- thisRep
  save.image("spines_model_t15_q1m.RData")
}
```

Extract data

```
newlist<-(mcmc.PALMSSpines.time[1:100])

unconstr <- list()
for (i in seq(1,100,1)){
  unconstr[[i]] <- newlist[[i]]
}

unconstr1b<-Filter(function(x) {nrow(x) >= 100}, unconstr)
length(unconstr1b)

summ_uc <- matrix(ncol=9, nrow=length(unconstr)*length(10:100))
colnames(summ_uc) <- names(unconstr[[1]])

for (par in 1:9){
  allRep<- NULL
  for (rep in 1:length(unconstr)){
    thisRep <- unconstr[[rep]][par][10:100,]
    allRep <- append(allRep, thisRep)
```

```

}
summ_uc[,par] <- allRep
}; rm("par", "rep", "thisRep", "allRep")
summ_uc <- data.frame(summ_uc)

```

Calculate speciation, extinction, transition and net diversification rates given constraints in model

```

summ_uc$lambda0.1<-summ_uc$lambda0.3
summ_uc$lambda1.1<-summ_uc$lambda0.3
summ_uc$lambda1.3<-summ_uc$lambda0.3
summ_uc$mu0.1<-summ_uc$mu0.3
summ_uc$mu1.1<-summ_uc$mu1.3
summ_uc$mu1.2<-summ_uc$mu0.3
summ_uc$mu0.2<-summ_uc$mu0.3
summ_uc$r0.1=summ_uc$lambda0.1-summ_uc$mu0.1
summ_uc$r0.2=summ_uc$lambda0.2-summ_uc$mu0.2
summ_uc$r0.3=summ_uc$lambda0.3-summ_uc$mu0.3
summ_uc$r1.1=summ_uc$lambda1.1-summ_uc$mu1.1
summ_uc$r1.2=summ_uc$lambda1.2-summ_uc$mu1.2
summ_uc$r1.3=summ_uc$lambda1.3-summ_uc$mu1.3
summ_uc$q01.1<-summ_uc$q01.3
summ_uc$q10.1<-summ_uc$q10.3
summ_uc$q01.2<-summ_uc$q10.3
summ_uc$q10.2<-summ_uc$q10.3

```

Save dataframe

```
write.csv(summ_uc, file="mcmc.PALMSspines_t15_q1m_100trees.csv")
```

Plot densities of rates

```

par(mfrow=c(2,2))
col <- c("#004165", "#eaab00", "red", "pink", "black", "grey")
profiles.plot(summ_uc[c("lambda0.1", "lambda0.2", "lambda0.3", "lambda1.1", "lambda1.2", "lambda1.3")],
profiles.plot(summ_uc[c("q01.1", "q01.2", "q01.3", "q10.1", "q10.2", "q10.3")], col.line=col, las=1,xlab="Divergence")
profiles.plot(summ_uc[c("r0.1", "r0.2", "r0.3", "r1.1", "r1.2", "r1.3")], col.line=col, las=1,xlab="Divergence")

```

Plot boxplots - prepare data

```

combi<-summ_uc

str(combi)
combi$X<- NULL
combi$i <- NULL
combi$p<- NULL

# lambda
lambda<-combi
lambda$mu0.1<- NULL
lambda$mu0.2<- NULL
lambda$mu0.3<- NULL
lambda$mu1.1<- NULL

```

```

lambda$mu1.2<- NULL
lambda$mu1.3<- NULL
lambda$q01.1 <- NULL
lambda$q10.1 <- NULL
lambda$q01.2 <- NULL
lambda$q10.2 <- NULL
lambda$q01.3 <- NULL
lambda$q10.3 <- NULL
lambda$r0.1<-NULL
lambda$r0.2<-NULL
lambda$r0.3<-NULL
lambda$r1.1<-NULL
lambda$r1.2<-NULL
lambda$r1.3<-NULL

```

```

# mu

```

```

mu<-combi
mu$lambda0.1<- NULL
mu$lambda0.2<- NULL
mu$lambda0.3<- NULL
mu$lambda1.1<- NULL
mu$lambda1.2<- NULL
mu$lambda1.3<- NULL
mu$q01.1 <- NULL
mu$q10.1 <- NULL
mu$q01.2 <- NULL
mu$q10.2 <- NULL
mu$q01.3 <- NULL
mu$q10.3 <- NULL
mu$r0.1<-NULL
mu$r0.2<-NULL
mu$r0.3<-NULL
mu$r1.1<-NULL
mu$r1.2<-NULL
mu$r1.3<-NULL

```

```

# q

```

```

q<-combi
q$lambda0.1<- NULL
q$lambda1.1 <- NULL
q$lambda0.2<- NULL
q$lambda1.2 <- NULL
q$lambda0.3<- NULL
q$lambda1.3 <- NULL
q$mu0.1<- NULL
q$mu0.2<- NULL
q$mu0.3<- NULL
q$mu1.1<- NULL
q$mu1.2<- NULL
q$mu1.3<- NULL
q$r0.1<-NULL
q$r0.2<-NULL
q$r0.3<-NULL
q$r1.1<-NULL

```

```
q$r1.2<-NULL
q$r1.3<-NULL
```

```
# r
r<-combi
r$lambda0.1<- NULL
r$lambda1.1 <- NULL
r$lambda0.2<- NULL
r$lambda1.2 <- NULL
r$lambda0.3<- NULL
r$lambda1.3 <- NULL
r$mu0.1<- NULL
r$q01.1 <- NULL
r$q10.1 <- NULL
r$q01.2 <- NULL
r$q10.2 <- NULL
r$q01.3 <- NULL
r$q10.3 <- NULL
r$mu0.1<- NULL
r$mu0.2<- NULL
r$mu0.3<- NULL
r$mu1.1<- NULL
r$mu1.2<- NULL
r$mu1.3<- NULL
```

```
# stack data (reshape)
stacked.data1 = melt(lambda)
stacked.data2 = melt(mu)
stacked.data3 = melt(q)
stacked.data4 = melt(r)
```

Plot and save as pdf

```
par(mfrow=c(2,2))
```

```
pdf("armature_boxplot_rates.pdf")
boxplots.triple = boxplot((value)~ variable, data = stacked.data1, cex=0.8, las=2, cex.lab=0.2)
boxplots.triple = boxplot((value)~ variable, data = stacked.data2, cex=0.8, las=2, cex.lab=0.2)
boxplots.triple = boxplot((value)~ variable, data = stacked.data3, cex=0.8, las=2, cex.lab=0.2)
boxplots.triple = boxplot((value)~ variable, data = stacked.data4, cex=0.8, las=2, cex.lab=0.2)
dev.off()
```