

```

1  set.seed(2024)
2
3  library("Claddis")
4  library("dplyr")
5  library("dispRity")
6  library("ggplot2")
7  library("nlme")
8  library("ape")
9  library("paleotree")
10 library("paleoTS")
11 library("phytools")
12 library("strap")
13 library("fBasics")
14 library("MuMIn")
15 library("MASS")
16 library("geiger")
17 library("geomorph")
18 library("vegan")
19 library("ggpubr")
20 library("rgl")
21 library("geometry")
22 library("webshot2")
23
24 #Load strict consensus tree
25 tree <- read.tree("abelis_SRCT.nex")
26 windows()
27 plot(tree)
28 #Read temporal data
29 ages <- read.table("FADLADalltax.txt", row.names = 1, sep = "\t", header = TRUE)
30
31 #Adjust taxon lists
32 equals <- intersect(row.names(ages), tree$tip.label)
33 temp_data.red <- ages[equals,]
34
35 #Generate temporal slices using stages
36 time.bins.intervals <- read.table("time_intervals.txt", row.names = 1, sep = "\t",
37 header = TRUE)
38
39 #Generate list of temporal ranges and matrix of taxa-bins
40 ranges_matrices <- binTimeData(temp_data.red, int.times=time.bins.intervals)
41
42 #Generate different matrices through subsampling
43 seqLists <- seqTimeList(ranges_matrices, nruns=100)
44
45 #Calculate sampling rate
46 sampPars <- t(apply(seqLists$timeLists, function(x) optimPaleo(make_durationFreqDisc(x)
47 ))$par))
48 meanIntLen <- apply(seqLists$timeLists, function(x) mean(-apply(x[[1]], 1, diff)))
49 sampRate <- apply(1:length(meanIntLen), function(x) sProb2sRate(R=sampPars[x, 2],
50 int.length=meanIntLen[x]))
51 sampRate
52
53 #Delete infinite rates
54 sRateClean <- sampRate[!is.infinite(sampRate)]
55 nTaxaClean <- seqLists$nTaxa[!is.infinite(sampRate)]
56 nIntClean <- seqLists$nIntervals[!is.infinite(sampRate)]
57 meanIntLenClean <- meanIntLen[!is.infinite(sampRate)]
58 windows()
59 boxplot(sRateClean, ylab="Est. Sampling Rate per Lmy (Inf Results Dropped)")
60 mtext(text=paste("Median:", round(median(sRateClean), 3)), 1, 2)
61
62 #Compare adjustment of different distributions
63 windows()
64 barplot(as.vector(Weights(c(
65   IC(fitdistr(sRateClean, "gamma")$loglik, 2, n=length(sRateClean)),
66   IC(fitdistr(sRateClean, "exponential")$loglik, 1, n=length(sRateClean)),
67   IC(fitdistr(sRateClean, "log-normal")$loglik, 2, n=length(sRateClean))
68 ))), names=c("gamma", "exponential", "log-normal"), ylab="Akaike Weights")
69
70 #Log-normal distribution shows the best fit
71 modelFit.samp <- fitdistr(sRateClean, "log-normal")$estimate

```

```

69
70 #Calculate extinction rates
71 extRate <- sampPars[,1]/meanIntLen
72 extClean <- extRate[!is.infinite(sampRate)]
73 median(extClean)
74
75 #Compare adjustment of different distributions
76 windows()
77 barplot(as.vector(Weights(c(
78   IC(fitdistr(extClean,"gamma")$loglik,2,n=length(extClean)),
79   IC(fitdistr(extClean,"exponential")$loglik,1,n=length(extClean)),
80   IC(fitdistr(extClean,"log-normal")$loglik,2,n=length(extClean))
81 ))),names=c("gamma","exponential","log-normal"),ylab="Akaike Weights")
82
83 #Gamma distribution shows the best fit
84 modelFit.ext <- fitdistr(extClean,"gamma")$estimate
85
86 #Simulate and plot both rates
87 windows()
88 layout(1:2)
89 plot(sRateClean,extClean,xlab="Sampling Rate",ylab="Extinction Rate")
90 sampRateSim <- rexp(10,rate=modelFit.samp)
91 extRateSim <- rexp(10,rate=modelFit.ext)
92 windows()
93 plot(sampRateSim,extRateSim,xlab="Simulated Sampling Rate",ylab="Simulated Extinction
Rate")
94
95 #Proceed with calibration
96 windows()
97 time.trees_cal3 <- list()
98 for (j in 1:length(sRateClean)) {
99   time.trees_cal3[[j]] <- cal3TimePaleoPhy(tree, temp_data.red,
100                                           ntrees = 10,
101                                           anc.wt = 0,
102                                           brRate = extClean[j],
103                                           extRate = extClean[j],
104                                           sampRate = sRateClean[j],
105                                           plot = T)
106 }
107
108 #Combine all trees into a single list
109 time.trees_cal3_set <- list()
110 for (i in 1:length(sRateClean)) {
111   time.trees_cal3_set <- c(time.trees_cal3_set, time.trees_cal3[[i]])
112 }
113
114 #Plot one of the trees as example
115 windows()
116 geoscalePhylo(ladderize(time.trees_cal3_set[[2]], right = T), ages = temp_data.red,
x.lim = c(0, 300), cex.tip = 0.9, cex.age = 0.5, cex.ts = 0.6,
label.offset = -1)
117
118
119 #Generate tree with mean ages
120 consesus.tree.cal3 <- consensus.edges(time.trees_cal3_set)
121 root.ages <- vector()
122 for (j in 1:length(time.trees_cal3_set)){
123   root.ages[j] <- time.trees_cal3_set[[j]]$root.time
124 }
125 mean.root.ages <- mean(root.ages)
126
127 consesus.tree.cal3$root.time <- mean.root.ages
128 windows()
129 geoscalePhylo(ladderize(consesus.tree.cal3, right = T), ages = temp_data.red, x.lim =
c(30, 250), cex.tip = 0.9, cex.age = 0.5, cex.ts = 0.6,
label.offset = -1)
130
131
132 consesus.tree.cal3.red <- drop.tip(consesus.tree.cal3,tip = c("Indosaurus","Afromimus"
))
133
134 #Transform any zero-length branch to 0.1
135 consesus.tree.cal3.red$edge.length[consesus.tree.cal3.red$edge.length == 0] <- 0.01

```

```
136
137 windows()
138 geoscalePhylo(ladderize(consesus.tree.cal3.red, right = T), ages = temp_data.red,
139 x.lim = c(30, 250), cex.tip = 0.9, cex.age = 0.5, cex.ts = 0.6,
140 label.offset = -1)
141 write.tree(consesus.tree.cal3.red)
142
```