Macropodoid Modelling

29/07/2019

Contents

Getting the Data Together

```
## Warning: package 'RPANDA' was built under R version 3.5.2
```

```
## Warning: package 'picante' was built under R version 3.5.2
```

```
## Warning: package 'ape' was built under R version 3.5.2
```

```
## Warning: package 'vegan' was built under R version 3.5.2
```

```
## Warning: package 'permute' was built under R version 3.5.2
```

```
## Warning: package 'nlme' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'geiger' was built under R version 3.5.2
```

```
## Warning: package 'mvtnorm' was built under R version 3.5.2
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

Now read in the tree files we'll be working with

```r
#min.tree <- read.nexus("/Users/Ian/Google.Drive/ANU Herp Work/Macropod_Dating/FossilUncertainty/Macro_
#mean.tree <- read.nexus("/PATH/Macro_MeanAges_CON.tre")
#max.tree <- read.nexus("/PATH/Macro_MaxAges_CON.tre")
#cp.min <- read.nexus("/PATH/Macro_CP_MinAges_CON.tre")
#cp.mean <- read.nexus("/PATH/Macro_CP_EstAges_CON.tre")
#cp.max <- read.nexus("/PATH/Macro_CP_MaxAges_CON.tre")
sampled.trees   <- read.tree("~/Google.Drive/ANU/Macropod_Dating/REAL_Run5_AllSchemes_508trees_Macropodi
empirical.trees <- read.tree("~/Google.Drive/ANU/Macropod_Dating/MODEL110_Sampled_Run2_Macropodinae.tre
fossil.trees    <- read.tree("~/Google.Drive/ANU/Macropod_Dating/REAL_Run4_Fossil_519trees_Macropodinae
consensus.tree <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Run5_C
```

Choose the current tree we want to work with

```r
tree <- consensus.tree
```

And the hypsodonty data

```r
# raw data for all the macropodoid taxa
all.HI           <- read.csv("~/Google.Drive/ANU/Macropod_Dating/FossilUncertainty/Data/CrownHeight_Data
# or just species means for the Macropodinae
hypsodonty.index <- read.csv("~/Google.Drive/ANU/Macropod_Dating/FossilUncertainty/Data/Macropodinae_Hyp
head(hypsodonty.index)
```

```
##                 Taxon        HI             HI_Source Diet_Guild
## 1 Baringa_nelsonensis 1.3259480 Couzens & Prideaux 2018    Browser
## 2     Baringa_sp_indet 1.1395682 Couzens & Prideaux 2018    Browser
## 3       Bohra_bandharr 1.0119760 Couzens & Prideaux 2018    Unknown
## 4     Bohra_illuminata 0.9373134 Couzens & Prideaux 2018    Unknown
## 5     Bohra_nullarbora 1.0722913 Couzens & Prideaux 2018    Unknown
## 6       Bohra_sp_indet 1.2234637 Couzens & Prideaux 2018    Unknown
##   Guild_Source Alt_Diet_Guild    min_HI    max_HI   mean_HI
## 1  Dawson 2006           <NA> 1.1399594 1.4338624 1.3259480
## 2  Dawson 2006           <NA> 0.7931034 1.3394077 1.1395682
## 3         <NA>           <NA> 1.0119760 1.0119760 1.0119760
## 4         <NA>           <NA> 0.9373134 0.9373134 0.9373134
## 5         <NA>           <NA> 0.9965096 1.1480730 1.0722913
## 6         <NA>           <NA> 1.2234637 1.2234637 1.2234637
##   lognormal_mean_error lognormal_sd_error
## 1           0.25580763         0.11399677
## 2           0.05568965         0.24949454
## 3           0.01190490                 NA
## 4          -0.06473755                 NA
## 5           0.06730667         0.07058468
## 6           0.20168592                 NA
```

The C4 plant reconstruction data from Andrae

```r
enviro.data <- read.csv("~/Google.Drive/ANU/Macropod_Dating/Andrae_S1.csv", header=T)
  grass.data <- enviro.data[1:25, c(1,3)]
  # this will also remove the last sample (from 9.5 mya) which is spurious, and keep just the mean esti
head(enviro.data); head(grass.data)
```

```
##   Age Age_Error C4_recon_mean C4_recon_lower C4_recon_upper
## 1 1.0     0.002          59.2           36.7           81.6
## 2 2.0     0.001          36.6           11.8           61.3
## 3 2.5     0.008          36.0           11.2           60.8
## 4 2.8     0.005          35.4           10.5           60.2
## 5 2.8     0.013          21.8            0.0           48.1
## 6 3.0     0.005          38.8           14.3           63.3
```

```
##   Age C4_recon_mean
## 1 1.0          59.2
## 2 2.0          36.6
## 3 2.5          36.0
## 4 2.8          35.4
## 5 2.8          21.8
## 6 3.0          38.8
```

And the dust flux data from Andrae

```r
flux.data <- read.csv("~/Google.Drive/ANU/Macropod_Dating/Aeolian_Flux.csv", header=T)
head(flux.data)
```

```
##           Age    A_Flux
## 1 0.05931646 108.3949
## 2 0.10874684 113.8622
## 3 0.20760760 118.6517
## 4 0.25703797 121.9088
## 5 0.35589873 109.4722
## 6 0.40532911 116.3859
```

As well as the paleotemperature data

```r
data(InfTemp)
head(InfTemp)
```

```
##      Age Temperature
## 1 0.000    3.902176
## 2 0.000    2.900296
## 3 0.002    4.309984
## 4 0.002    5.172534
## 5 0.004    3.733446
## 6 0.004    4.309984
```

Trim tree and data down to overlapping taxa

```r
# extract the taxa that are in both the tree and
overlaps <- intersect(tree$tip.label, unique(hypsodonty.index$Taxon))
macro.tree <- drop.tip(tree, setdiff(tree$tip.label, overlaps))
#macro.tree <- lapply(tree, drop.tip, tip=tip.drops) # if you're using a set of trees (fossil, sampled)
trim.data <- dplyr::filter(hypsodonty.index, Taxon %in% overlaps)
macro.HI <- trim.data[,2]; names(macro.HI) <- trim.data[,1]; geiger::name.check(macro.tree, macro.HI)
```

```
## [1] "OK"
```

```r
macro.HI
```

```
##          Baringa_nelsonensis              Bohra_illuminata
##                    1.3259480                     0.9373134
##      Dendrolagus_bennettianus          Dendrolagus_dorianus
##                    1.0115887                     0.9400000
##      Dendrolagus_goodfellowi          Dendrolagus_inustus
##                    0.8500000                     0.8899909
##        Dendrolagus_lumholtzi         Dendrolagus_matschiei
##                    0.9342720                     0.9100000
##              Dorcopsis_hageni              Dorcopsis_veterum
##                    0.8800000                     1.0200000
##          Dorcopsoides_fossilis          Dorcopsulus_vanheurni
##                    0.7554314                     0.9793040
##              Kurrabi_mahoneyi  Lagorchestes_conspicillatus
##                    1.4331210                     1.1378107
##          Lagorchestes_hirsutus         Lagostrophus_fasciatus
##                    1.2047786                     1.1600000
##              Macropus_agilis          Macropus_antilopinus
##                    1.1692005                     1.2300000
##              Macropus_eugenii         Macropus_fuliginosus
##                    1.1109478                     1.3630443
```

```
##           Macropus_giganteus              Macropus_irma
##                  1.3300000                    1.1372544
##              Macropus_parma               Macropus_parryi
##                  1.3176042                    1.3475744
##             Macropus_pavana             Macropus_robustus
##                  1.1823511                    1.2418003
##         Macropus_rufogriseus               Macropus_rufus
##                  1.3500000                    1.3978825
##         Onychogalea_fraenata        Onychogalea_unguifera
##                  1.5000000                    1.2694731
##           Peradorcas_concinna           Petrogale_assimilis
##                  1.2222222                    1.1667479
##         Petrogale_brachyotis            Petrogale_inornata
##                  1.0074257                    1.6500000
##           Petrogale_lateralis         Petrogale_penicillata
##                  1.2076257                    1.2858835
##       Petrogale_purpureicollis         Petrogale_rothschildi
##                  1.8200000                    1.4000000
##         Petrogale_xanthopus Prionotemnus_palankarinnicus
##                  1.2100000                    1.0221543
##             Protemnodon_anak            Setonix_brachyurus
##                  1.0869565                    1.0196716
##         Thylogale_billardierii             Thylogale_brunii
##                  1.1231331                    1.0000000
##         Thylogale_stigmatica             Thylogale_thetis
##                  1.1895715                    1.1652379
##             Wallabia_bicolor
##                  1.1218826
```

If you're working with the raw data, trim tree and data down to just Macropodinae

```r
macros <- dplyr::filter(all.HI, Higher_tax == "Macropodinae")
overlaps <- intersect(tree$tip.label, unique(all.HI$Taxon))
trim.tree <- drop.tip(tree, setdiff(tree$tip.label, overlaps))
trim.raw <- filter(macros, Taxon %in% overlaps)
```

create a tibble to get the species means (if you haven't done this already)

```r
library(dplyr)
sp.means <- trim.raw %>%
  group_by(Taxon) %>%
  summarise_at(vars(H_HYPCD/PW), mean)
#write.csv(sp.means, row.names=FALSE, file="/PATH/CrownHeight_Macropodinae_spMEANS.csv") # uncomment
```

Visualizing Our Data

Let's quickly visualize the data in a few different ways to get an idea of what's going on

```r
# Barplot of trait value
plotTree.barplot(macro.tree, macro.HI, args.barplot=list(beside=TRUE, border=F))
```
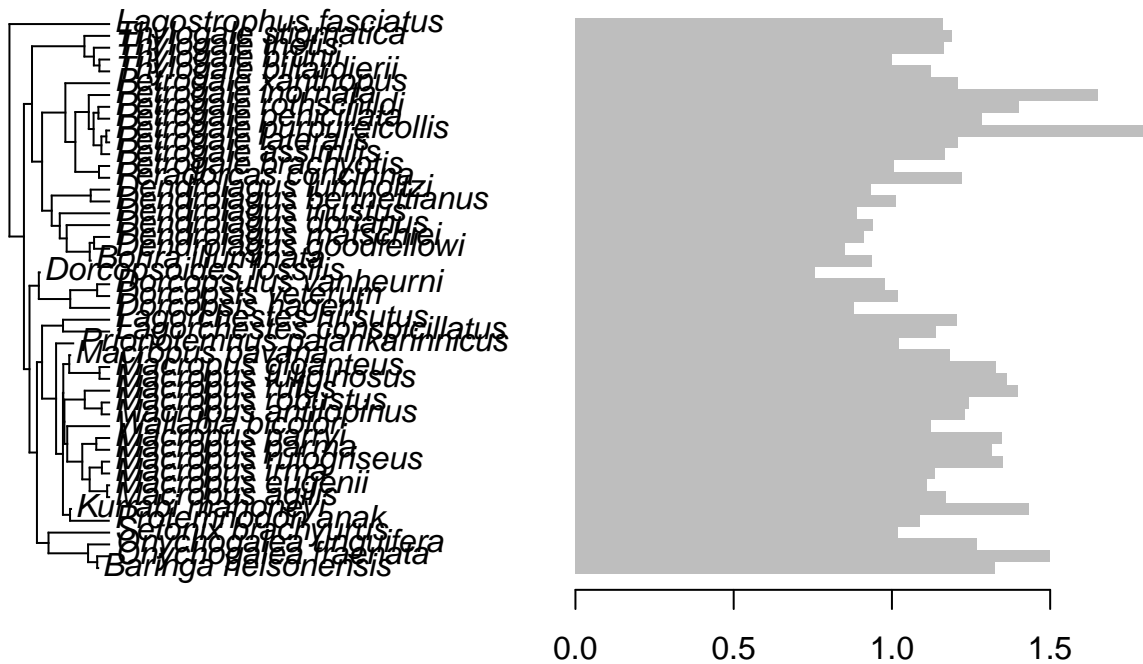
```
# Continuous trait map
obj1 <- contMap(macro.tree, macro.HI, plot=FALSE, outline=F);
    n<-length(obj1$cols);
    obj1$cols[1:n] <- rev(colorRampPalette(brewer.pal(9, "YlGnBu"))(n));
plot(obj1,legend=0.7*max(nodeHeights(obj1$tree)),
     fsize=c(0.7,0.9), lwd=5, border=F); axisPhylo(1, backward=T)
```
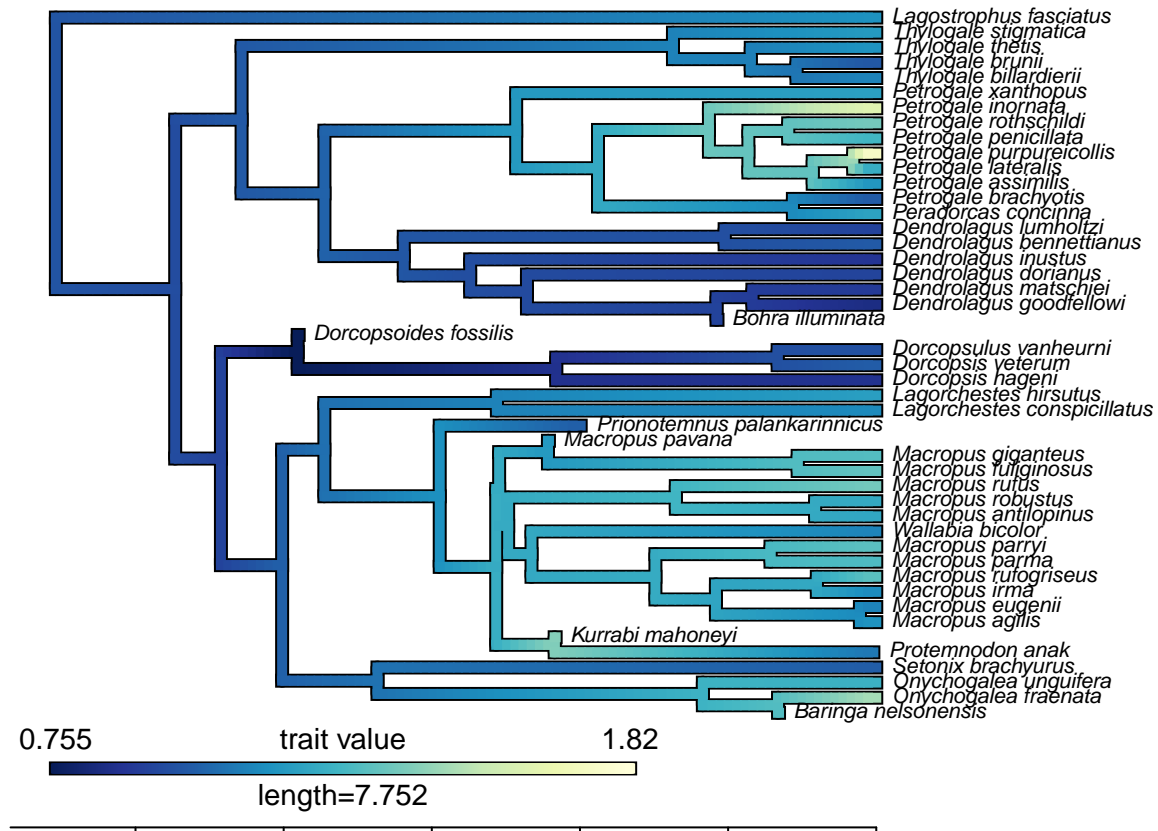
```
# Hypsodonty distributed across feeding guilds
hidata <- hypsodonty.index[complete.cases(hypsodonty.index[,c(1,2,4)]),]
    hidata <- hidata[order(hidata$HI),]
        hidata$Taxon <- factor(hidata$Taxon, levels = hidata$Taxon)
ggplot(hidata, aes(x=Taxon, y=HI, fill=Diet_Guild), colour=brewer.pal(3,"Paired")) +
  geom_col() + theme_classic() + coord_flip() +
  scale_x_discrete(limits = rev(hidata$Taxon)) +
  scale_fill_brewer(palette="Paired", "Diet_Guild")
```
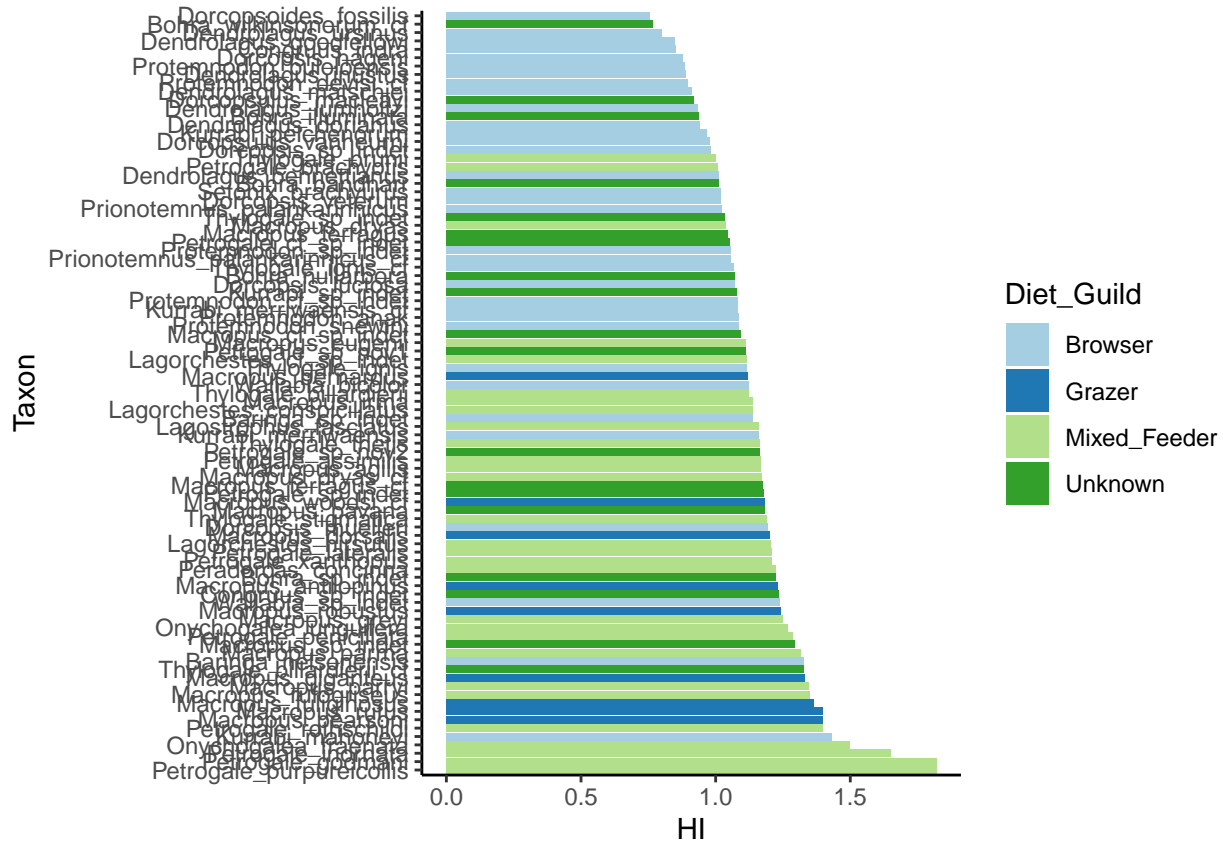

Now we can look at the time-sampled variables

```
# make a plot of the grass data
pp <- ggplot(enviro.data[1:25,], aes(Age)) +
  geom_ribbon(aes(ymin = C4_recon_lower, ymax = C4_recon_upper), fill = "light green") +
  geom_line(aes(y = C4_recon_mean), color="DarkGreen") + scale_x_reverse() + theme_classic() +
  coord_cartesian(xlim = c(0, 20), ylim = c(0,80), expand = FALSE)
#qq <- gggeo_scale(pp, dat="epochs") # if you want to plot a geological timescale

# make a plot of the flux data
rr <- ggplot(flux.data, aes(Age)) +
  geom_ribbon(aes(ymin = A_Flux-35, ymax = A_Flux+35), fill = "light blue") +
  geom_line(aes(y = A_Flux), color="DarkBlue") + scale_x_reverse() + theme_classic() +
  coord_cartesian(xlim = c(0, 20), ylim = c(0,150), expand = FALSE)
#ss <- gggeo_scale(rr, dat="epochs") # if you want to plot a geological timescale

grid.arrange(pp, rr, nrow=1)
```
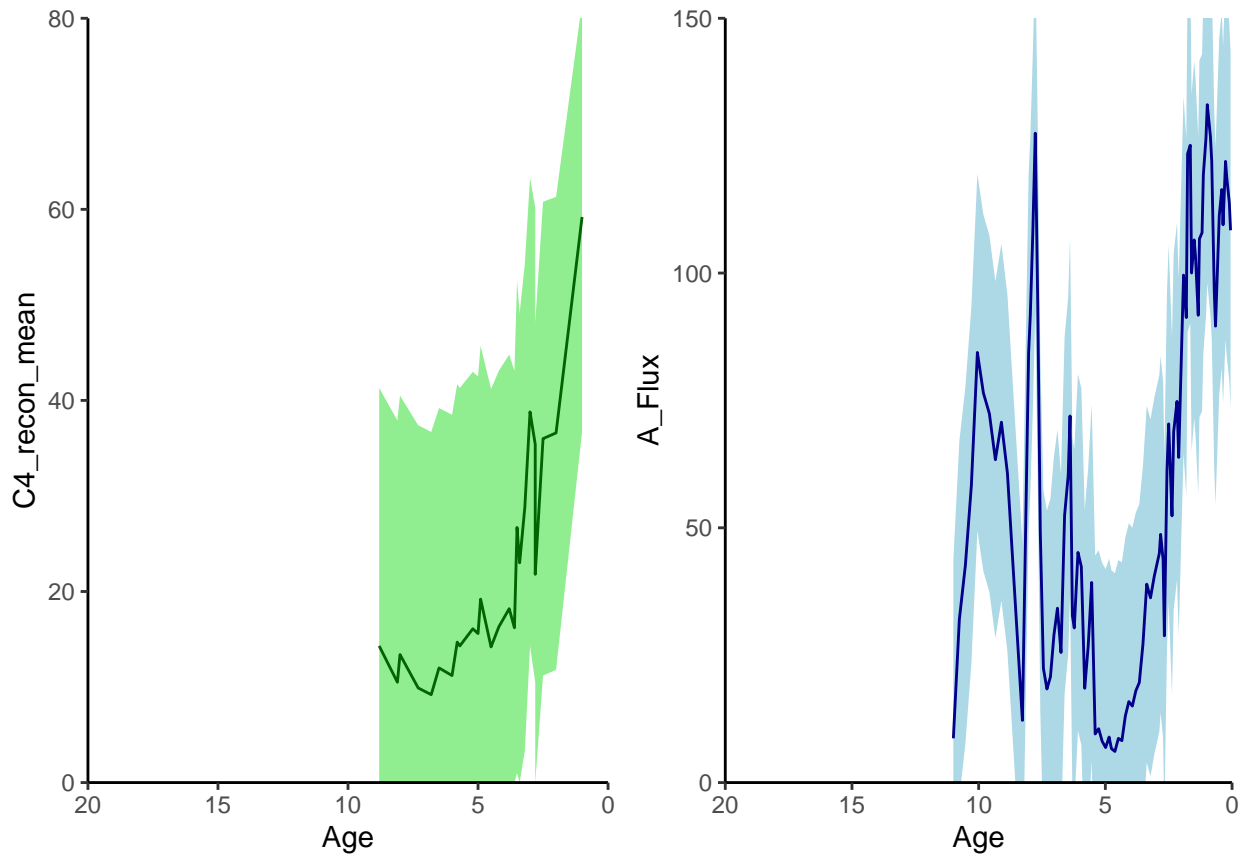
Fitting Models of Trait Evolution

Correlative models like the environmental models in RPANDA will be sensitive to the amount of smoothing to the trend line of the input data (see Clavel & Morlon, PNAS). To address this, we'll create a function that searches for the optimum smoothness of the trend by fitting a set of values.

```
best.smoothing <- function (phy, trait.data, time.data=InfTemp,
                            degrees=c(0,10,20,30,40,50), model="EnvExp", cores=6) {
  res.list <- mclapply(1:length(degrees), function(x) {
    fit_t_env(phy, trait.data, env_data=time.data, df=degrees[x],
            scale=F, plot=T, model=model)}, mc.cores = cores)
  for(i in 1:length(res.list)){res.list[[i]]$df <- degrees[i]}
  res.values <- unlist(lapply(res.list, function(x) x$aicc)) # make a vector of the values, so we can g
  best.res <- res.list[[which.min(res.values)]]

  plot(best.res, main=paste(model, "; AICc = ", round(best.res$aicc,2)),
       sub=paste("sigma = ",round(best.res$param[1],2),
                " beta = ",round(best.res$param[2],2),
                " df = ", best.res$df), col="red")

  return(list(all.results=res.list, best.result=best.res, best.df=best.res$df))
}
```

We can have a look at what this smoothing actually does to our data. We can come back to look at these once we get the optimum fits for our models.

```
grass.spline0  <- sm.spline(x=grass.data$Age, y=grass.data$C4_recon_mean, df=0)
grass.spline10 <- sm.spline(x=grass.data$Age, y=grass.data$C4_recon_mean, df=10)
grass.spline20 <- sm.spline(x=grass.data$Age, y=grass.data$C4_recon_mean, df=20)
grass.spline30 <- sm.spline(x=grass.data$Age, y=grass.data$C4_recon_mean, df=30)
grass.spline40 <- sm.spline(x=grass.data$Age, y=grass.data$C4_recon_mean, df=40)
grass.spline50 <- sm.spline(x=grass.data$Age, y=grass.data$C4_recon_mean, df=50)

plot(grass.data, main="C4 Grass Reconstruction Through Time", xlab="Millions of Years Ago", ylab="Percer
lines(grass.spline0, col="red", lwd=4)
lines(grass.spline10, col="green", lwd=4)
lines(grass.spline20, col="blue", lwd=4)
lines(grass.spline30, col="yellow", lwd=4)
lines(grass.spline40, col="violet", lwd=4)
lines(grass.spline50, col="gold", lwd=4)
```

## C4 Grass Reconstruction Through Time
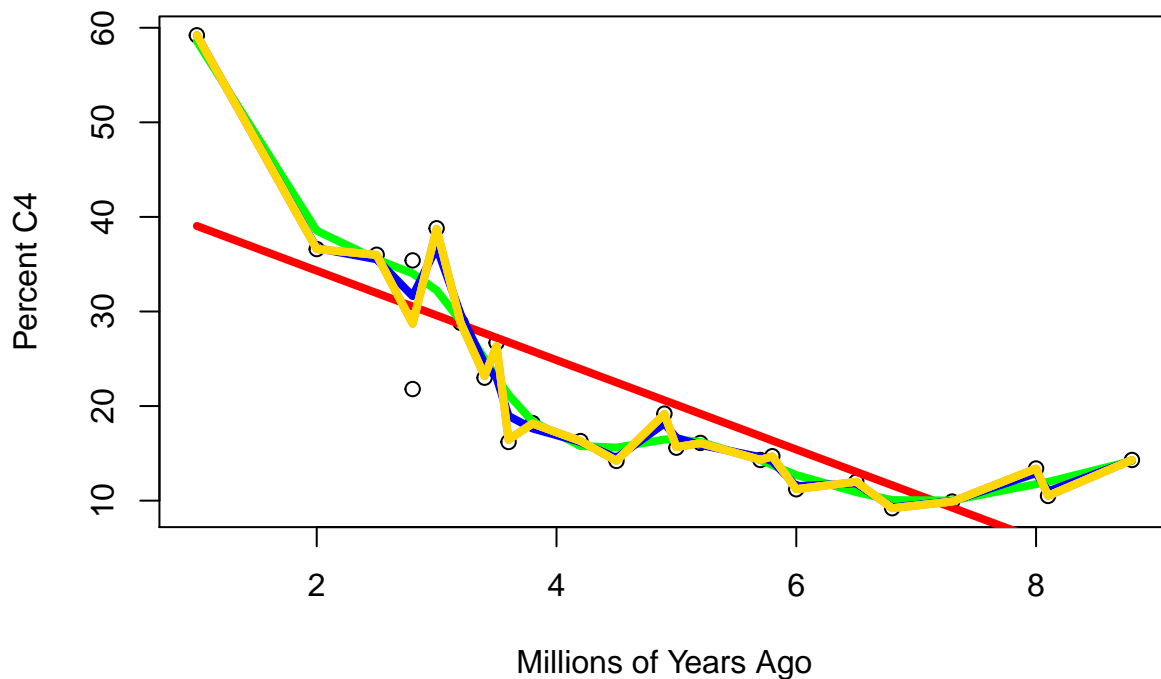

```
flux.spline0  <- sm.spline(x=flux.data$Age, y=flux.data$A_Flux, df=0)
flux.spline10 <- sm.spline(x=flux.data$Age, y=flux.data$A_Flux, df=10)
flux.spline20 <- sm.spline(x=flux.data$Age, y=flux.data$A_Flux, df=20)
flux.spline30 <- sm.spline(x=flux.data$Age, y=flux.data$A_Flux, df=30)
flux.spline40 <- sm.spline(x=flux.data$Age, y=flux.data$A_Flux, df=40)
flux.spline50 <- sm.spline(x=flux.data$Age, y=flux.data$A_Flux, df=50)

plot(flux.data, main="Aeolian Dust Flux Through Time", xlab="Millions of Years Ago", ylab="Aeolian Dust
lines(flux.spline0, col="red", lwd=4)
lines(flux.spline10, col="green", lwd=4)
lines(flux.spline20, col="blue", lwd=4)
lines(flux.spline30, col="yellow", lwd=4)
lines(flux.spline40, col="magenta", lwd=4)
lines(flux.spline50, col="cyan", lwd=4)
```

Aeolian Dust Flux Through Time
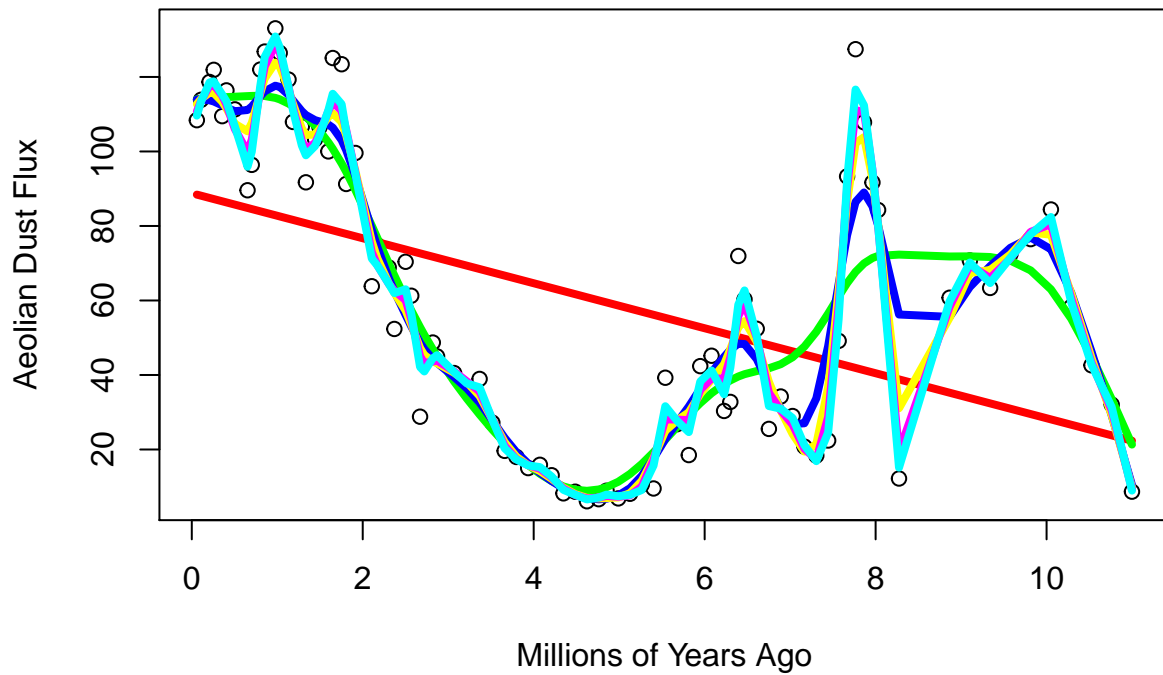

```
data(InfTemp)
env.spline0 <- sm.spline(x=InfTemp$Age, y=InfTemp$Temperature, df=0)
env.spline10 <- sm.spline(x=InfTemp$Age, y=InfTemp$Temperature, df=10)
env.spline20 <- sm.spline(x=InfTemp$Age, y=InfTemp$Temperature, df=20)
env.spline30 <- sm.spline(x=InfTemp$Age, y=InfTemp$Temperature, df=30)
env.spline40 <- sm.spline(x=InfTemp$Age, y=InfTemp$Temperature, df=40)
env.spline50 <- sm.spline(x=InfTemp$Age, y=InfTemp$Temperature, df=50)

plot(InfTemp, main="Paleotemperature Through Time")
lines(env.spline0, col="red")
lines(env.spline10, col="green")
lines(env.spline20, col="blue")
lines(env.spline30, col="yellow")
lines(env.spline40, col="magenta")
lines(env.spline50, col="cyan")
```

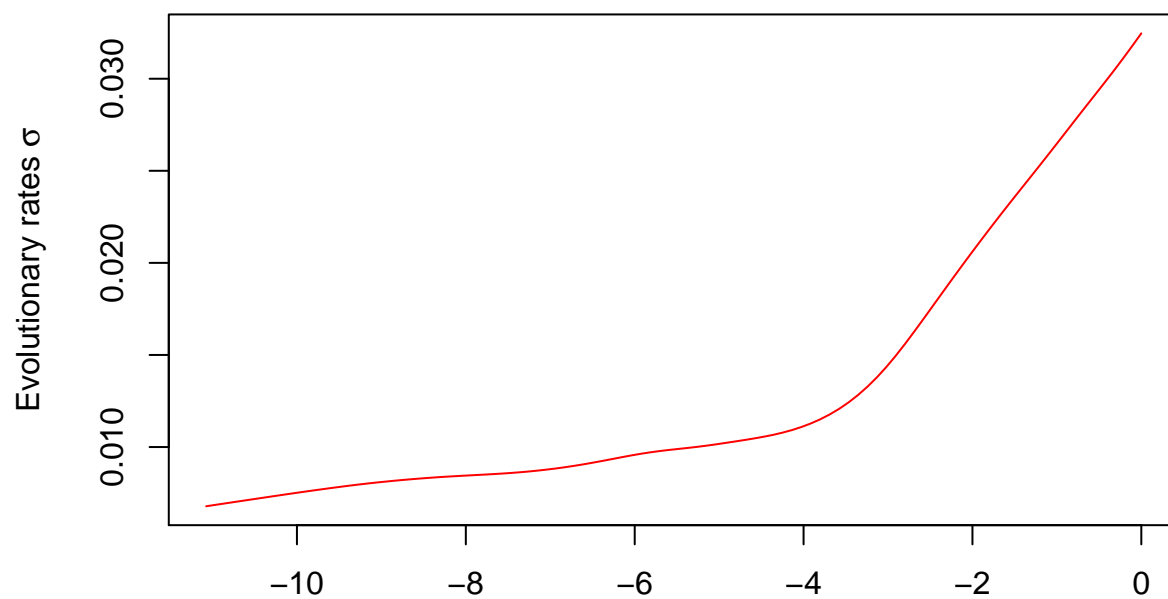**Paleotemperature Through Time**
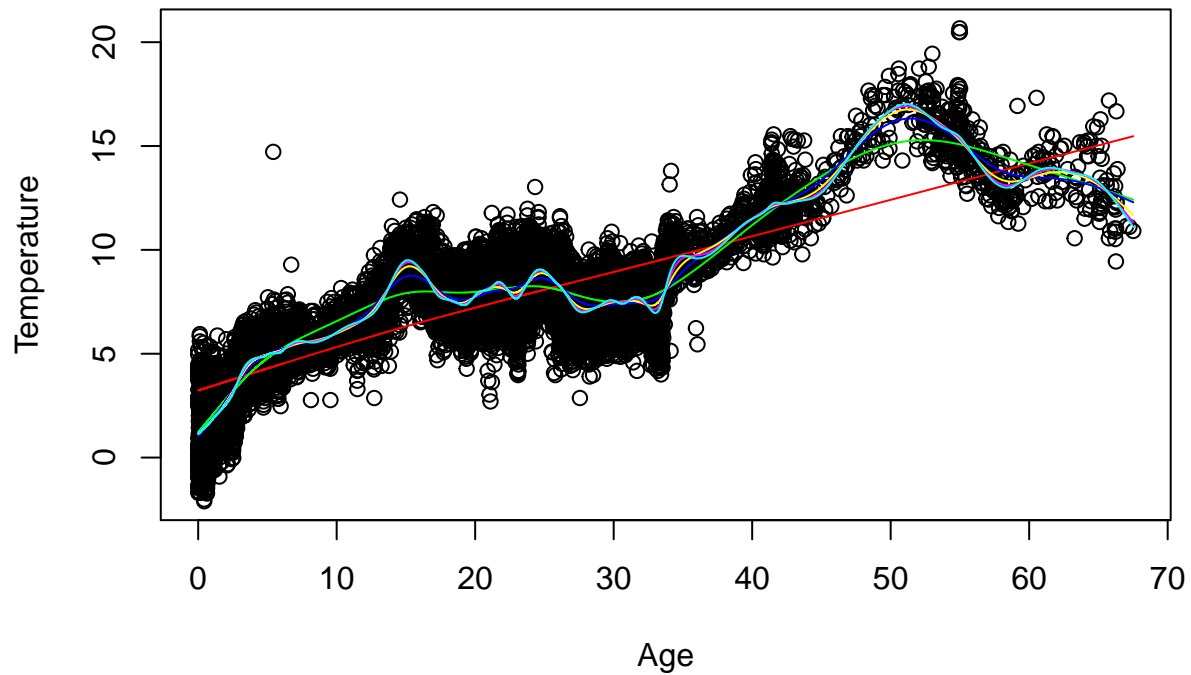
Next we'll fit a number of models to our tree and data

Start with the environmental model of paleotemperature. You can designate the number of cores and the amount of smoothing

```
# linear model first
ENVexp <- best.smoothing(macro.tree, macro.HI, time.data=InfTemp,
                         degrees=c(10,20,30,40,50), model="EnvExp", cores=5)
```
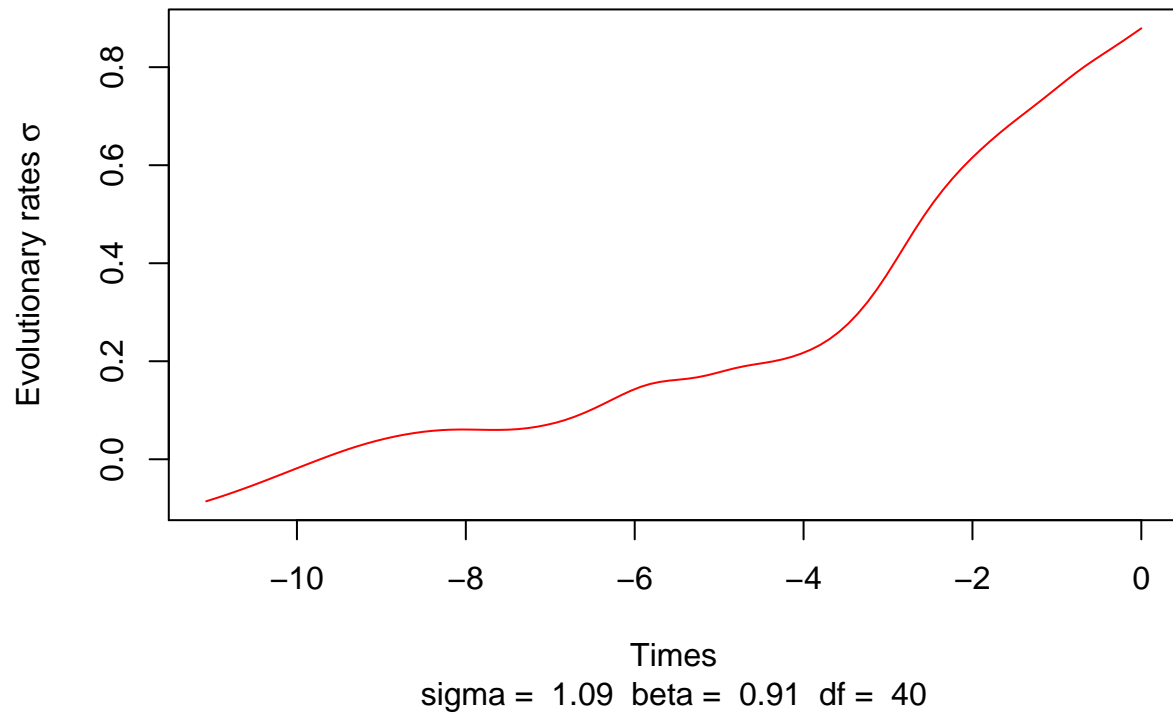
EnvExp ; AICc = 14.21

Times
sigma = 0.05 beta = −0.3 df = 30

```
# exponential model next
ENVlin <- best.smoothing(macro.tree, macro.HI, time.data=InfTemp,
                         degrees=c(10,20,30,40,50), model="EnvLin", cores=5)
```
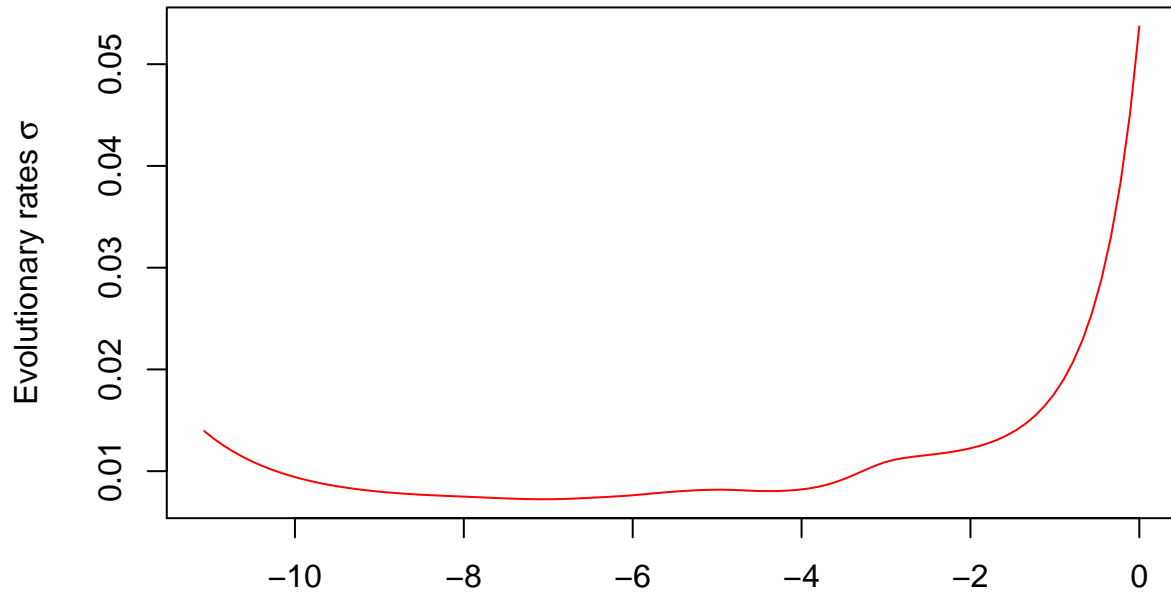
EnvLin ; AICc = 82.08

sigma = 1.09 beta = 0.91 df = 40

Next up the grass model, using C4 reconstructions.

```
GRASSexp <- best.smoothing(macro.tree, macro.HI, time.data=grass.data,
                           degrees=c(10,20,30,40,50), model="EnvExp", cores=5)
```
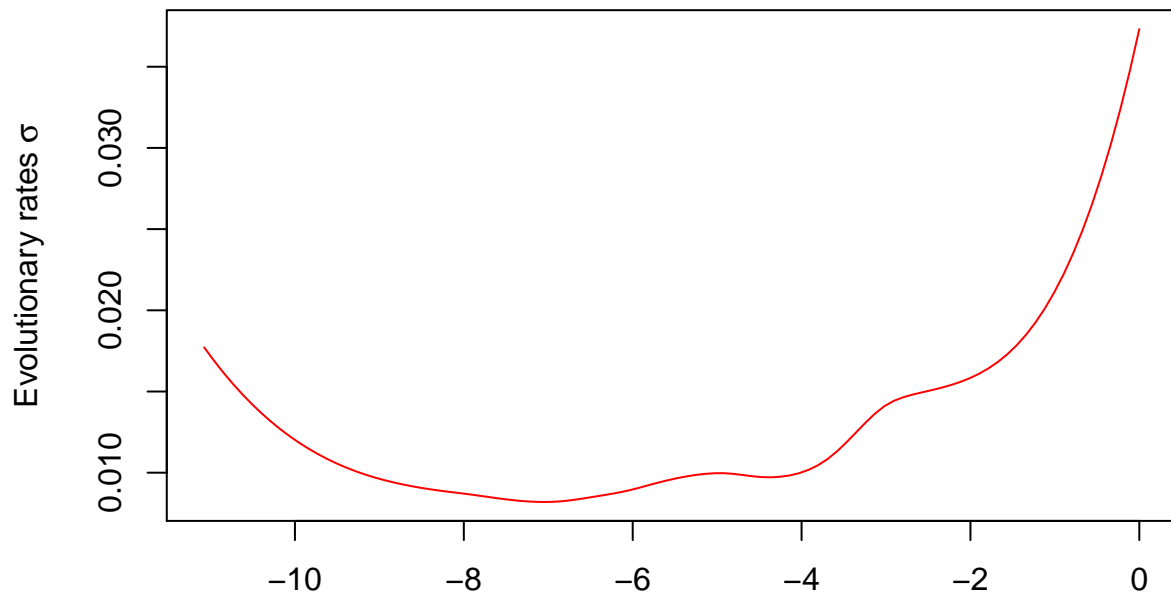
EnvExp ; AICc = 6.72

sigma = 0.01 beta = 0.02 df = 10

```
GRASSlin <- best.smoothing(macro.tree, macro.HI, time.data=grass.data,
                           degrees=c(10,20,30,40,50), model="EnvLin", cores=5)
```
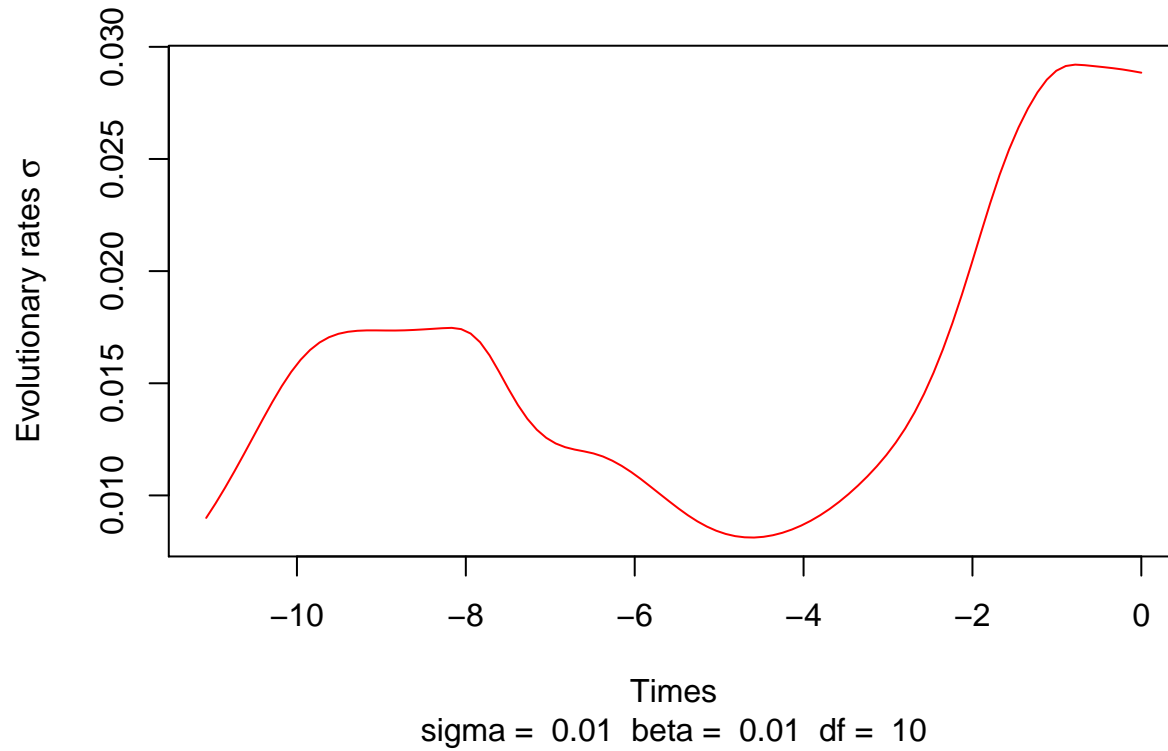
EnvLin ; AICc = 10.42

sigma = 0.01 beta = 0.01 df = 10

And finally the flux models, using aeolian dust measurements.
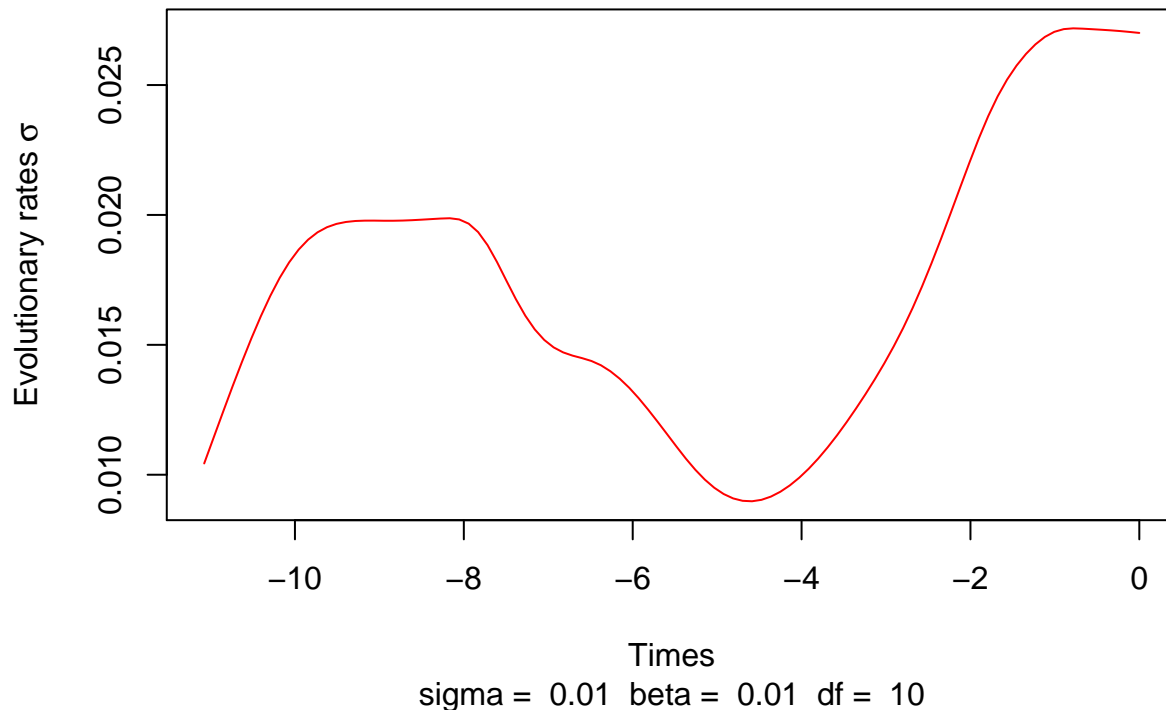
```
FLUXexp <- best.smoothing(macro.tree, macro.HI, time.data=flux.data,
                          degrees=c(10,20,30,40,50), model="EnvExp", cores=5)
```


EnvExp ; AICc = 15.26

```
FLUXlin <- best.smoothing(macro.tree, macro.HI, time.data=flux.data,
                          degrees=c(10,20,30,40,50), model="EnvLin", cores=5)
```

EnvLin ; AICc = 16.34

sigma = 0.01 beta = 0.01 df = 10

Lastly, for comparison, run a few standard models. These are Brownian Motion, Brownian Motion with a Trend, and Early Burst.

```
BM_res <-    fitContinuous(macro.tree, macro.HI, model="BM")
trend_res <- fitContinuous(macro.tree, macro.HI, model="trend")
EB_res <-    fitContinuous(macro.tree, macro.HI, model="EB")
```

```
## Warning in fitContinuous(macro.tree, macro.HI, model = "EB"): Parameter estimates appear at bounds:
##  a
```

Compare the models with AICc, and check differences across the trees

```
model_FIT <- c(ENVexp$best.result$aicc, ENVlin$best.result$aicc,
               GRASSlin$best.result$aicc, GRASSexp$best.result$aicc,
               FLUXexp$best.result$aicc, FLUXlin$best.result$aicc,
               BM_res$opt$aicc, trend_res$opt$aicc, EB_res$opt$aicc);
names(model_FIT) <- c("ENVexp", "ENVlin", "GRASSlin", "GRASSexp", "FLUXexp", "FLUXlin", "BM", "Trend",
aic.w(model_FIT)
```

```
##      ENVexp      ENVlin    GRASSlin    GRASSexp     FLUXexp     FLUXlin
## 0.01949302 0.00000000 0.12961408 0.82510494 0.01154649 0.00672102
##          BM       Trend          EB
## 0.00192663 0.00497934 0.00061449
```

```
fit.aic <- as.data.frame(as.vector(aic.w(model_FIT)));
    fit.aic$model <- names(model_FIT);
        colnames(fit.aic) <- c("aiccw", "model");
            fit.aic$age <- "Consensus Tree"
```

Quickly collapse models from the same data

```
fit.aic$model.type <- c("ENV", "ENV", "GRASS", "GRASS",
                        "FLUX", "FLUX", "BM", "Trend", "EB")
fit.aic$model.type <- factor(fit.aic$model.type,
                        levels=c("ENV", "FLUX", "GRASS", "BM", "EB", "Trend"))
```

Then plot the model fits as AICc weights

```
library(wesanderson)
(ggplot(fit.aic)
  + geom_bar(aes(y=aiccw, x=age, fill=model.type), stat="identity")
  #+ theme(axis.text.x=element_text(angle=25, hjust=1), panel.background=element_blank(), legend.positi
  + theme_classic()
  + scale_fill_manual( values=wes_palette("Zissou1", 6, "continuous")))
```
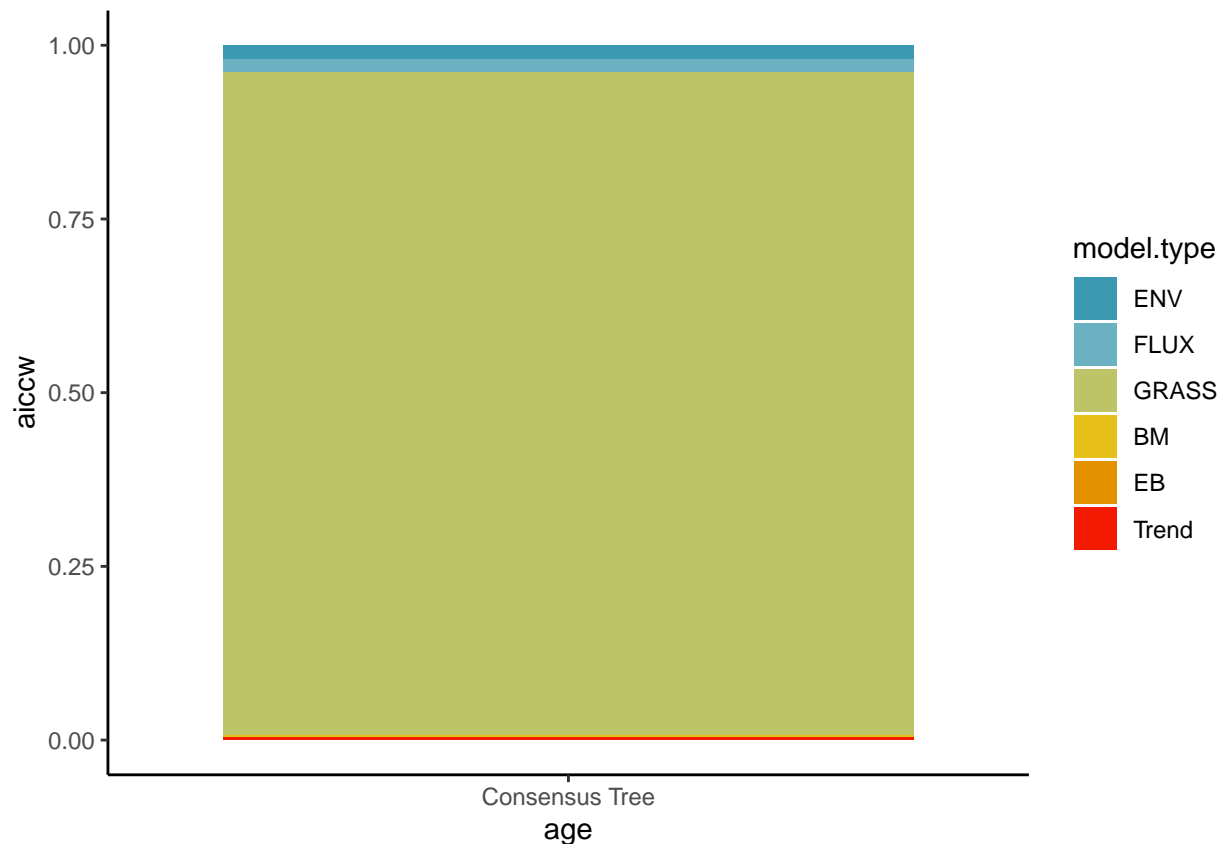

Fitting Models to Our Data as a Function of Time

Ok, now that we've fit the models to a given tree, we want to fit the models to lots of trees of different ages, shapes, etc. Now we'll run a loop across all of these trees, to fit the models to each one. It may take a little while.

```
# MAKE THIS EVAL=TRUE IF YOU WANT TO DO THIS BIT FOR REAL
tree.span <- fossil.trees
mean.data <- fossil.HI
# Fit all the models to a series of trees!
all.aics <- NULL; all.results <- NULL; timer <- progress_estimated(length(tree.span))
for (k in 1:length(tree.span)){
```

```r
int.results <- NULL

# Fit a number of models to the data (ENV, GRASS, BM, EB, Trend, Drift)
ENVexp <- best.smoothing(tree.span[[k]], mean.data, time.data=InfTemp,
                          degrees=c(10,20,30,40,50), model="EnvExp", cores=5);
                          int.results[["ENVexp"]] <- ENVexp$best.result;
ENVlin <- best.smoothing(tree.span[[k]], mean.data, time.data=InfTemp,
                          degrees=c(10,20,30,40,50), model="EnvLin", cores=5);
                          int.results[["ENVlin"]] <- ENVlin$best.result;

GRASSexp <- best.smoothing(tree.span[[k]], mean.data, time.data=grass.data,
                            degrees=c(30,40,50), model="EnvExp", cores=3);
                            int.results[["GRASSexp"]] <- GRASSexp$best.result;
GRASSlin <- best.smoothing(tree.span[[k]], mean.data, time.data=grass.data,
                            degrees=c(30,40,50), model="EnvLin", cores=3);
                            int.results[["GRASSlin"]] <- GRASSlin$best.result;

FLUXexp <- best.smoothing(tree.span[[k]], mean.data, time.data=flux.data,
                           degrees=c(10,20,30,40,50), model="EnvExp", cores=5);
                           int.results[["FLUXexp"]] <- FLUXexp$best.result;
FLUXlin <- best.smoothing(tree.span[[k]], mean.data, time.data=flux.data,
                           degrees=c(10,20,30,40,50), model="EnvLin", cores=5);
                           int.results[["FLUXlin"]] <- FLUXlin$best.result;

BM_res    <- fitContinuous(tree.span[[k]], mean.data, model="BM");
            int.results[["BM"]] <- BM_res
trend_res <- fitContinuous(tree.span[[k]], mean.data, model="trend");
            int.results[["Trend"]] <- trend_res
EB_res    <- fitContinuous(tree.span[[k]], mean.data, model="EB");
            int.results[["EB"]] <- EB_res

curr_tree_FIT <- c(ENVexp$best.result$aicc, ENVlin$best.result$aicc,
                   GRASSexp$best.result$aicc, GRASSlin$best.result$aicc,
                   FLUXexp$best.result$aicc, FLUXlin$best.result$aicc,
                   BM_res$opt$aicc, trend_res$opt$aicc, EB_res$opt$aicc);
names(curr_tree_FIT) <- c("ENVexp", "ENVlin", "GRASSexp", "GRASSlin",
                          "FLUXexp", "FLUXlin", "BM", "Trend", "EB")

curr_tree_SIG <- c(ENVexp$best.result$param[[1]], ENVlin$best.result$param[[1]],
                   GRASSexp$best.result$param[[1]], GRASSlin$best.result$param[[1]],
                   FLUXexp$best.result$param[[1]], FLUXlin$best.result$param[[1]],
                   BM_res$opt$sigsq, trend_res$opt$sigsq, EB_res$opt$sigsq);
names(curr_tree_SIG) <- c("ENVexp", "ENVlin", "GRASSexp", "GRASSlin",
                          "FLUXexp", "FLUXlin", "BM", "Trend", "EB")

curr_tree_PAR <- c(ENVexp$best.result$param[[2]], ENVlin$best.result$param[[2]],
                   GRASSexp$best.result$param[[2]], GRASSlin$best.result$param[[2]],
                   FLUXexp$best.result$param[[2]], FLUXlin$best.result$param[[2]],
                   NA, trend_res$opt$slope, EB_res$opt$a);
names(curr_tree_PAR) <- c("ENVexp", "ENVlin", "GRASSexp", "GRASSlin",
                          "FLUXexp", "FLUXlin", "BM", "Trend", "EB")

curr.aic <- as.data.frame(as.vector(aic.w(curr_tree_FIT)));
```

```r
        curr.aic$model <- names(curr_tree_FIT); colnames(curr.aic) <- c("aiccw", "model");
          curr.aic$age <- round(max(nodeHeights(tree.span[[k]])), 4)
              curr.aic$tree <- k
                curr.aic$sigsq <- curr_tree_SIG
                    curr.aic$par <- curr_tree_PAR
  all.results[[k]] <- int.results

  curr.aic$model.type <- c("ENV", "ENV", "GRASS", "GRASS",
                        "FLUX", "FLUX", "BM", "Trend", "EB")

  all.aics <- rbind.data.frame(all.aics, curr.aic);

  print(timer$tick())

}
```

Save the file externally:

```r
saveRDS(all.aics,    file="/PATH/Fossil_Trees_Model_Fitting_AICCs.RDS")
saveRDS(all.results, file="/PATH/Fossil_Trees_Model_Fitting_Results.RDS")
#saveRDS(all.aics, file="/PATH/Model_Fitting_AICCs.RDS")
```

Or skip the work and read in the file instead:

```r
all.aics <- readRDS("~/Google.Drive/ANU/Macropod_Dating/FossilUncertainty/Data/FINAL_SAMPLED_Model_Fitt
all.aics$model.type <- factor(all.aics$model.type,
                      levels=c("ENV", "FLUX", "GRASS", "BM", "EB", "Trend"))
all.aics$age <- as.factor(all.aics$age)
```

```r
sampled.res <- (ggplot(all.aics)
  + geom_bar(aes(y=aiccw, x=age, fill=model.type), stat="identity")
  + theme(axis.text.x=element_text(angle=90, hjust=1),
          panel.background=element_blank(), legend.position="none")
  + scale_fill_manual( values=wes_palette("Zissou1", 6, "continuous")))
sampled.res
```
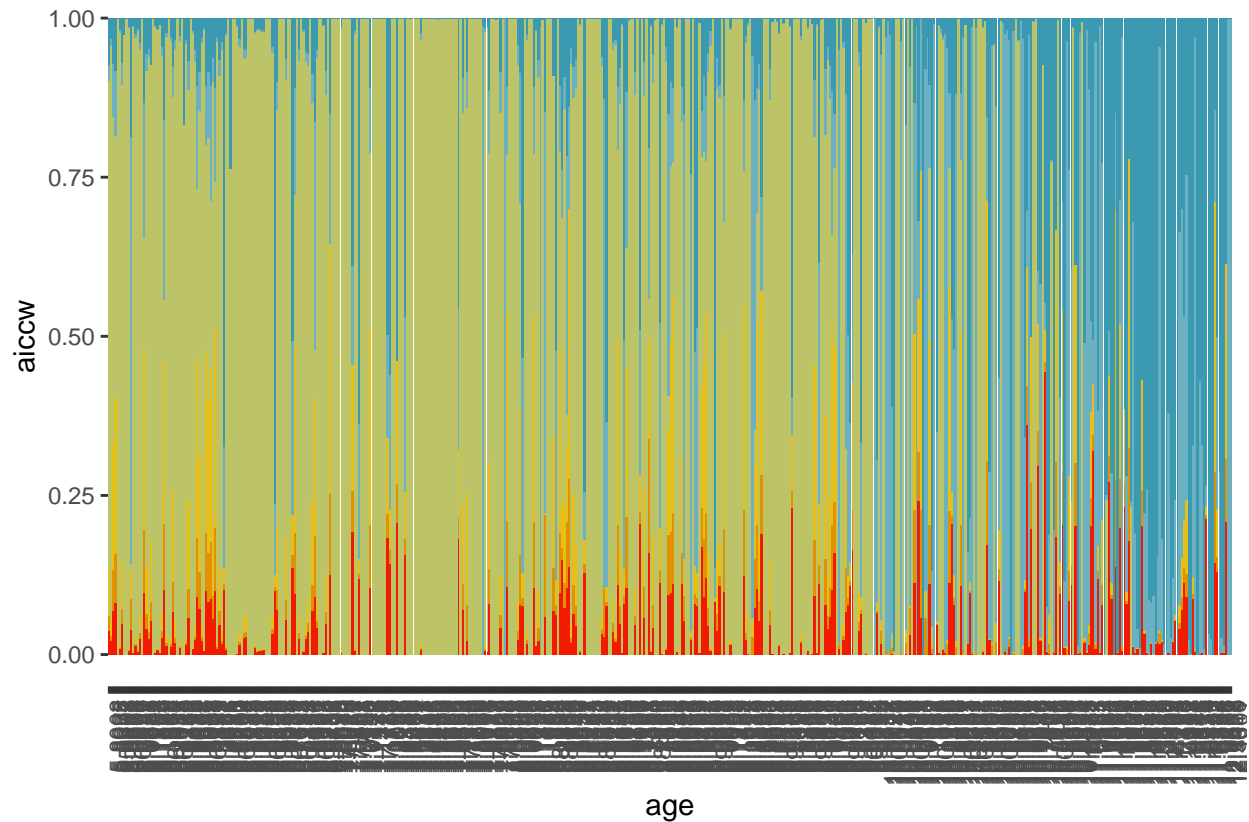
And then summarize the model support (average AICcWt) across all trees

```r
# Summarize the model support (average AICcWt)
aic.sum <- summarySE(all.aics, measurevar="aiccw", groupvars="model")
aic.sum$model <- factor(aic.sum$model, levels=c("ENVlin", "ENVexp", "FLUXlin", "FLUXexp", "GRASSexp", "G
aic.sum$model.type <- c("BM", "EB", "ENV", "ENV", "FLUX", "FLUX", "GRASS", "GRASS", "Trend")
aic.sum$model.type <- factor(aic.sum$model.type, levels=c("ENV", "FLUX", "GRASS", "BM", "EB", "Trend"))

fossil.bar <- (ggplot(aic.sum, aes(x=model.type, y=aiccw, fill=model.type))
  + geom_bar(stat="identity")
  #+ geom_errorbar(aes(ymin=aiccw-se, ymax=aiccw+se), size=0.3, width=0.2)
  + scale_fill_manual( values=wes_palette("Zissou1", 6, "continuous"))
  + theme_classic()
  + theme(legend.position="none")
  #+ facet_wrap(~group, nrow=3, ncol=2)
  + geom_text(aes(label=percent(aiccw), vjust=-4)))
fossil.bar
```
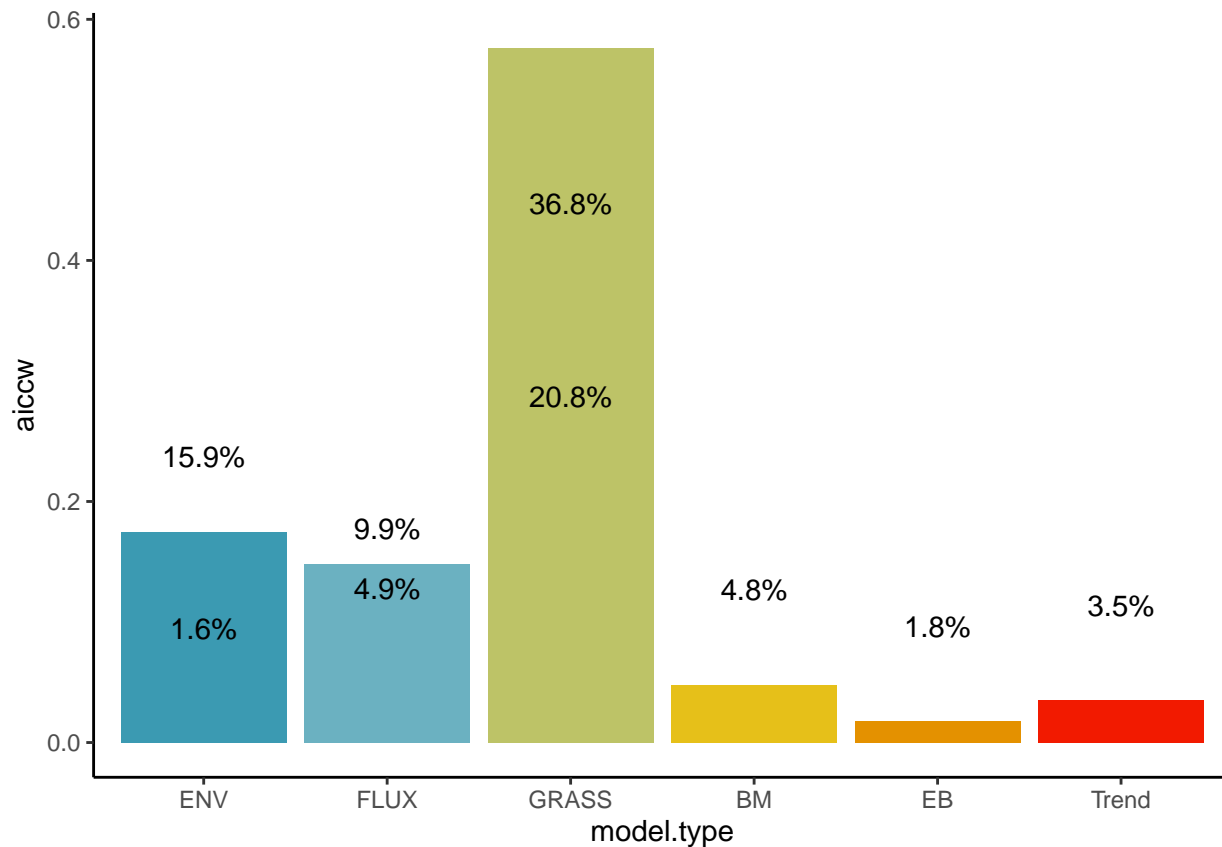
We can also visualize the estimated evolutionary rates of the trait.

First create an adjusted version of the plotting function from RPANDA:

```r
# Adjust the RPANDA plotting function so we can fix the axes, and do a bunch of plots
plot.fixed_t_env <- function (x, steps = 100,
                              xlim=c(-10,0), ylim=c(0,1), linecol="red", ...)
{
  if (is.function(x$model)) {
    fun_temp <- function(x, temp, model, param) {
      rate_fun <- function(x) {
        model(x, temp, param)
      }
      rate <- rate_fun(x)
      return(rate)
    }
  }
  else if (x$model == "EnvExp") {
    fun_temp <- function(x, temp, model, param) {
      sig <- param[1]
      beta <- param[2]
      rate <- (sig * exp(beta * temp(x)))
      return(rate)
    }
  }
  else if (x$model == "EnvLin") {
    fun_temp <- function(x, temp, model, param) {
      sig <- param[1]
```

```
    beta <- param[2]
    rate <- sig + (beta - sig) * temp(x)
    return(rate)
  }
}
t <- seq(0, x$tot_time, length.out = steps)
rates <- fun_temp(x = t, temp = x$env_func, model = x$model,
                  param = x$param)
plot(-t, rates, type = "l", xlab = "Times",
     ylab = bquote(paste("Evolutionary rates ",sigma)),
     xlim=xlim, ylim=ylim, col=linecol, ...)
results <- list(time_steps = t, rates = rates)
invisible(results)
}
```

I'll plot just the results from the GRASS-linear model, but you could do this for all.

```
require(dplyr)
all.aics <- readRDS("~/Google.Drive/ANU/Macropod_Dating/FossilUncertainty/Data/FINAL_SAMPLED_Model_Fitti
all.grass.results <- readRDS("~/Google.Drive/ANU/Macropod_Dating/Model_Fitting_Results/REAL_SAMPLED_GRAS
sub.aic <- filter(all.aics, model=="GRASSlin" & aiccw > 0.5); sub.trees <- sub.aic$tree; print(paste(le
```

```
## [1] "111 estimates"
```

```
for(i in sub.trees){plot.fixed_t_env(all.grass.results[[i]]$GRASSlin,
                                     xlim=c(-12,0), ylim=c(0,0.2),
                                     linecol=randomcoloR::randomColor(1));
  par(new=T)}
```
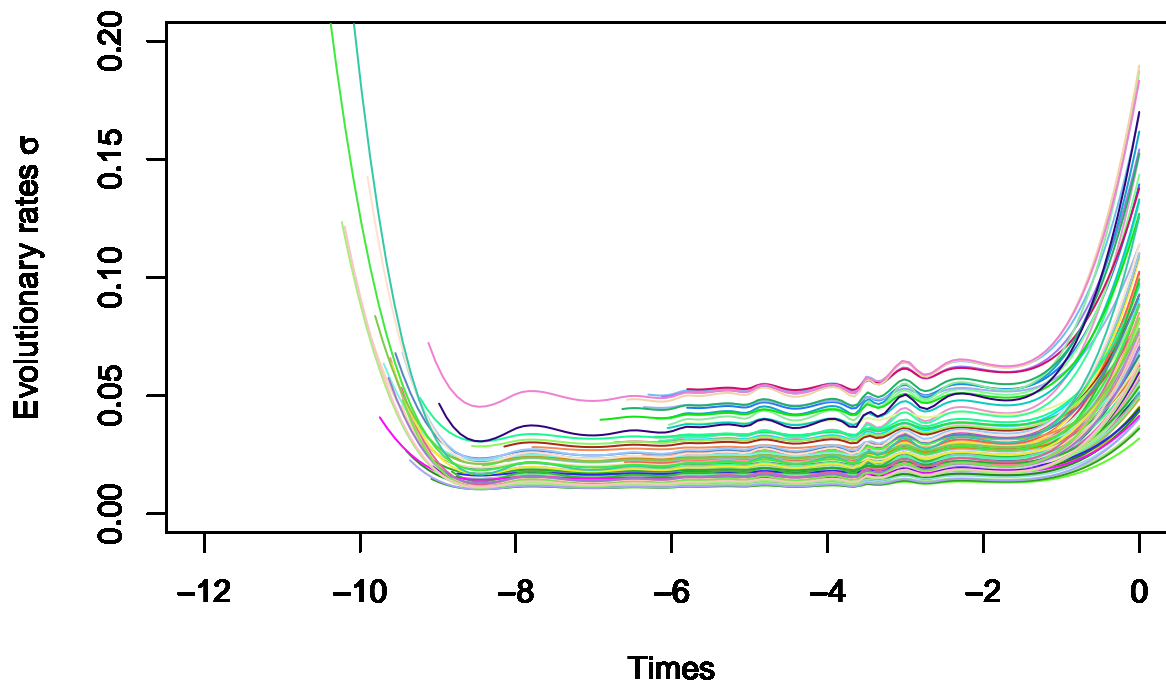

Also plot the beta value of the correlation between rate and the variable

```
# And plot the beta values for preferred models
col.pal <- brewer.pal(8, "Paired")
beta.values <- filter(all.aics, model %in% c("GRASSexp", "FLUXexp", "GRASSlin", "ENVexp") & aiccw >= 0.
```

```
ge_beta <- filter(beta.values, model=="GRASSexp");
ge <- ggplot(ge_beta, aes(x=model, y=par)) + geom_violin(fill=col.pal[[4]]) + theme_classic() + geom_bo:

gl_beta <- filter(beta.values, model=="GRASSlin");
gl <- ggplot(gl_beta, aes(x=model, y=par)) + geom_violin(fill=col.pal[[3]]) + theme_classic() + geom_bo:

fe_beta <- filter(beta.values, model=="FLUXexp" );
fe <- ggplot(fe_beta, aes(x=model, y=par)) + geom_violin(fill=col.pal[[1]]) + theme_classic() + geom_bo:

ee_beta <- filter(beta.values, model=="ENVexp"  );
ee <- ggplot(ee_beta, aes(x=model, y=par)) + geom_violin(fill=col.pal[[2]]) + theme_classic() + geom_bo:
gridExtra::grid.arrange(gl, ge, fe, ee, nrow=1)
```
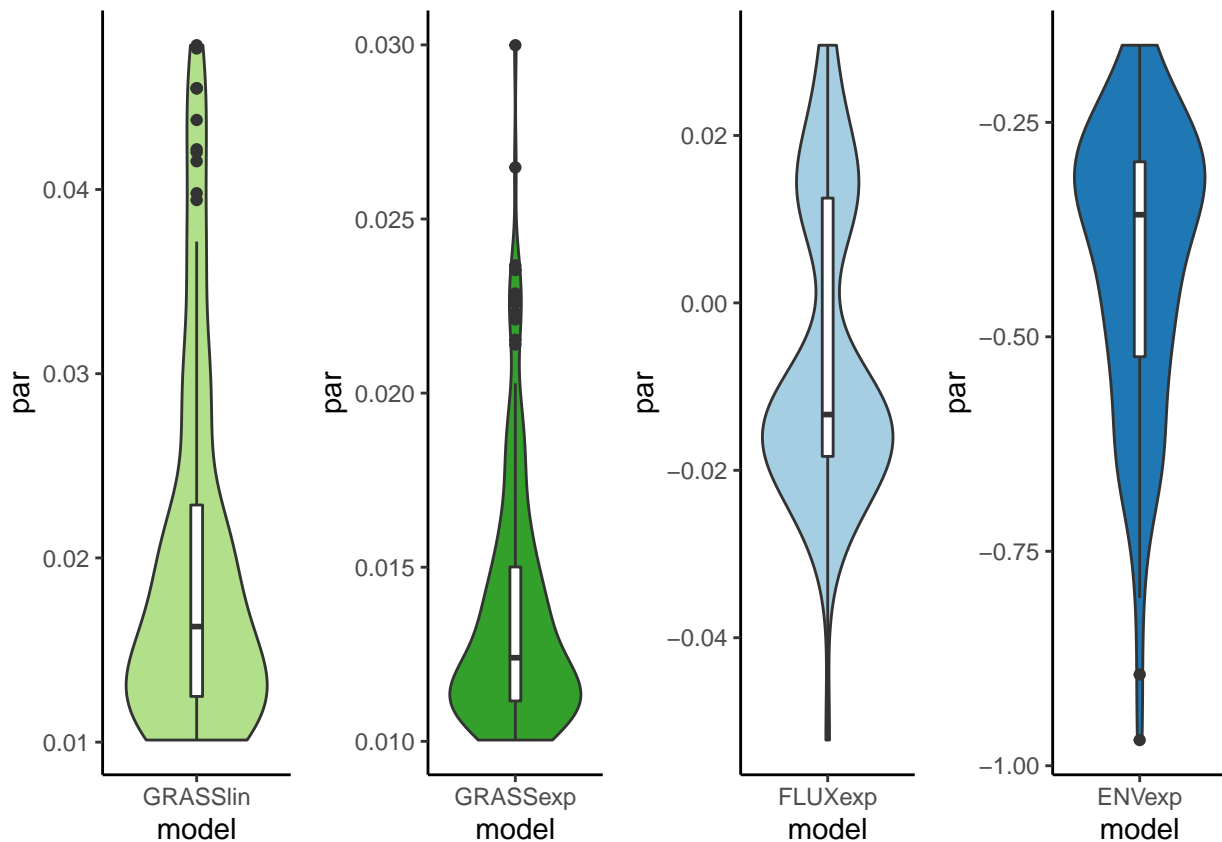

Comparing Estimated Node Ages Across Dating Schemes

Make a quick function to get the DEPTH of a node (from present), instead of the HEIGHT (from root)

```
MRCA.depth <- function(phy){max(nodeHeights(phy)) - findMRCA(phy, tips=c("Macropus_irma", "Wallabia_bico
```

Prepare trees for comparison of ages

```
# Prepare trees for comparison of ages
max.age.trees <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Alterna
max.ages <- max.age.trees[sample(1:length(max.age.trees),300)]
#max.ages <- lapply(max.age.trees, drop.tip, tip=setdiff(max.age.trees[[1]]$tip.label, overlaps)); clas
age.max <- as.data.frame(unlist(lapply(max.ages, MRCA.depth)));
```

```r
age.max$tree <- "fixed.max"; colnames(age.max) <- c("age", "tree")

mean.age.trees <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Alterna
mean.ages <- mean.age.trees[sample(1:length(mean.age.trees),300)]
#mean.ages <- lapply(mean.age.trees, drop.tip, tip=setdiff(mean.age.trees[[1]]$tip.label, overlaps)); c
age.mean <- as.data.frame(unlist(lapply(mean.ages, MRCA.depth)));
age.mean$tree <- "fixed.mean"; colnames(age.mean) <- c("age", "tree")

min.age.trees <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Alterna
#min.ages <- lapply(min.age.trees, drop.tip, tip=setdiff(min.age.trees[[1]]$tip.label, overlaps)); clas
age.min <- as.data.frame(unlist(lapply(min.ages, MRCA.depth)));
age.min$tree <- "fixed.min"; colnames(age.min) <- c("age", "tree")

sampled500.trees <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Run5_
sampled.ages <- sampled500.trees[sample(1:length(sampled500.trees),300)]
sampled500.trees <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Run5_
sampled.ages <- sampled500.trees[(length(sampled500.trees)-200):length(sampled500.trees)]
#min.ages <- lapply(min.age.trees, drop.tip, tip=setdiff(min.age.trees[[1]]$tip.label, overlaps)); clas
age.sampled <- as.data.frame(unlist(lapply(sampled.ages, MRCA.depth))); age.sampled$tree <- "est.prior"
```

Combine the ages from different dating schemes, and plot them

```r
require(wesanderson)
age.all <- rbind(age.min, age.sampled, age.mean, age.max)
age.all$tree <- factor(age.all$tree, levels=c("fixed.max","est.prior", "fixed.mean", "fixed.min"))
(ggplot(age.all, aes(x=age, fill=tree))
  + geom_density(alpha=0.75, adjust=1.5)
  + theme(axis.text.x=element_text(angle=0, hjust=1), panel.background=element_blank(), legend.position=
  + scale_fill_manual(values=wes_palette("Zissou1", type="continuous", 4))
  + scale_x_reverse(lim=c(8,0)))
```
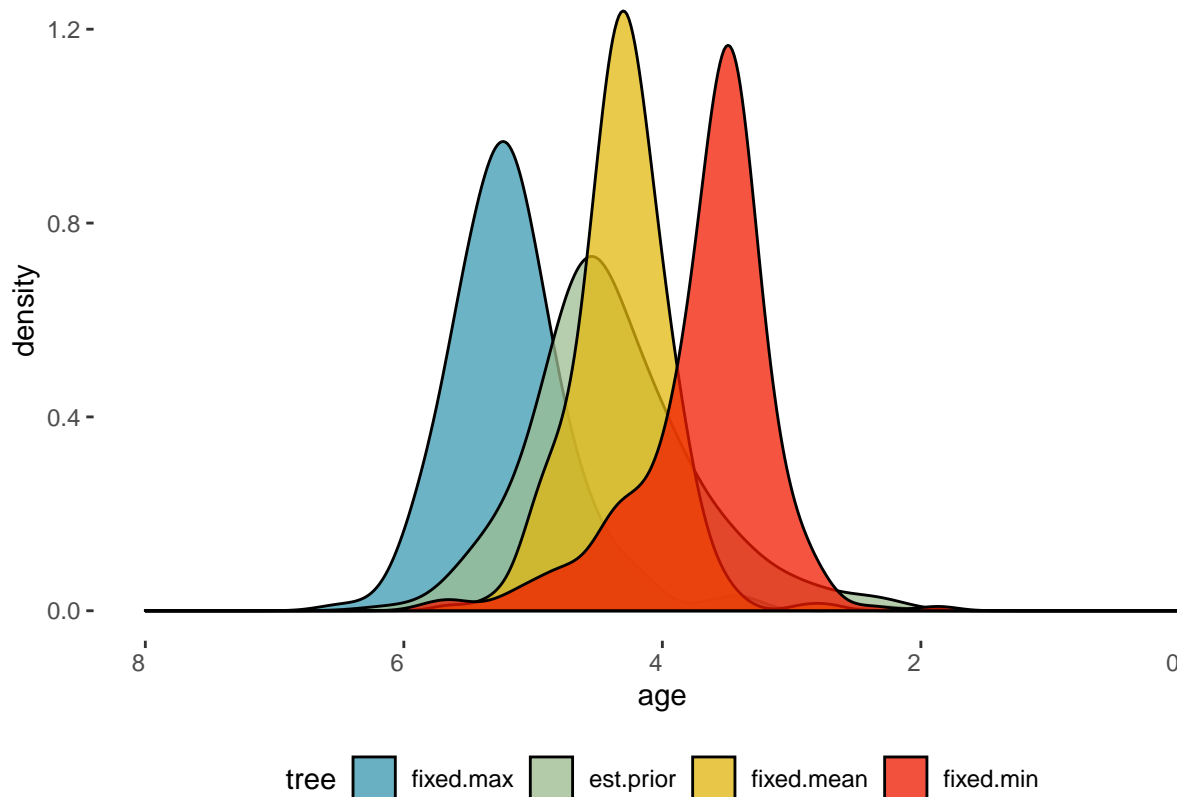
Getting Bayes Factors for Fossil Taxa

Investigating Fossil Taxa as putative Sampled Ancestors or Terminals

```
prior.trees <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Alternativ
prior.trees <- prior.trees[(length(prior.trees)-1000):length(prior.trees)]
post.trees  <- read.nexus("~/Google.Drive/ANU/Macropod_Dating/Operators/REAL_Macropodinae_ALL/Alternativ
post.trees <- post.trees[(length(post.trees)-1000):length(post.trees)]
```

Choose which tips you want information for:

```
fossil_taxa <- c("Baringa_nelsonensis","Congruus_congruus",
                 "Kurrabi_mahoneyi","Macropus_pavana")
```

Get the node numbers of the tips

```
nodes <- sapply(fossil_taxa,function(x,y) which(y==x),y=tree$tip.label)
```

Then get the edge lengths for those nodes

```
edge.lengths <- setNames(tree$edge.length[sapply(nodes,
                 function(x,y) which(y==x),y=tree$edge[,2])],names(nodes))
```

The faster way is to make a function to do this:

```
get_terminal_branchlengths <- function(phy, tipnames){
  ## Get the node numbers of the tips
  nodes <- sapply(tipnames,function(x,y) which(y==x),y=phy$tip.label)
  ## Then get the edge lengths for those nodes
```

```
    edge.lengths <- setNames(phy$edge.length[sapply(nodes,
                    function(x,y) which(y==x),y=phy$edge[,2])],names(nodes))
    return(edge.lengths)
}
```

Now that we've got the tips and branch lengths, we can compare the posterior to the prior

```
BFSA <- function(prior.phy, posterior.phy, tips){
  post <- lapply(posterior.phy, get_terminal_branchlengths, tipnames=tips);
      names(post) <- NULL; post <- unlist(post)
  prior <- lapply(prior.phy,    get_terminal_branchlengths, tipnames=tips);
      names(prior)<- NULL; prior <- unlist(prior)

  BFs <- NULL
  for (j in 1:length(tips)){
    curr.tip <- subset(post, names(post)==tips[j]);
    probSA <- sum(curr.tip<=0); probTIP <- length(curr.tip)-probSA;

    curr.tip <- subset(prior, names(prior)==tips[j]);
    priorSA <- sum(curr.tip<=0); priorTIP <- length(curr.tip)-priorSA;

    curr.BF <- log((probSA * priorTIP) / (probTIP * priorSA))
    if(is.na(curr.BF)){curr.BF <- 0}

    #curr.BF <- log(probSA/(length(curr.tip)-probSA))
    names(curr.BF) <- tips[j]; curr.BF <- round(curr.BF, 2)
    BFs <- append(BFs, curr.BF)
  }
  return(BFs)
}
```

Orient the data appropriately

```
macro_BFs <- BFSA(prior.trees, post.trees, tips=fossil_taxa)

macro_BFs <- as.data.frame(macro_BFs) # make the vector a data frame
macro_BFs[which(macro_BFs$macro_BFs > 5),] <- 5 # change any really big (INF) numbers to 5
macro_BFs[which(macro_BFs$macro_BFs < -5),] <- -5 # change any really small (-INF) numbers to -5

macro_BFs$taxa <- rownames(macro_BFs); # create column with the taxon names
macro_BFs <- macro_BFs[order(macro_BFs$macro_BFs),] # reorder by BF values
# set colors for plotting
macro_BFs$color <- "black";
#macro_BFs[which(macro_BFs$macro_BFs > 1),]$color <- "#3d98d3"
macro_BFs[which(macro_BFs$macro_BFs < -1),]$color <- "#FF7175"

macro_BFs$taxa <- factor(macro_BFs$taxa, levels=c(macro_BFs$taxa)) # set factors for plotting (NOT NECE
```

Then plot the data

```
ggplot(macro_BFs, aes(x=taxa, y=macro_BFs, label=macro_BFs)) +
  geom_ribbon(aes(ymin=-1, ymax=+1)) +
  geom_point(stat='identity', size=8, color=macro_BFs$color)  +
  # geom_segment(aes(y = 0,
  #                  x = taxa,
  #                  yend = macro_BFs,
```

```
#                   xend = taxa),
#              color = macro_BFs$color) +
geom_text(color="white", size=2) +
#labs(title="Bayes Factor Support", subtitle="for Fossil Taxa as Sampled Ancestors") +
#ylim(-5, 5) +
scale_y_continuous(name="log Bayes Factors", limits=c(-5,5), breaks=c(-5:5)) +
scale_x_discrete(limits = rev(unique(sort(macro_BFs$taxa)))) + # drop this if you want to order it di
#theme(panel.background=element_blank()) +
geom_hline(yintercept=-1) +
geom_hline(yintercept=1) +
xlab("Fossil Taxa") +
#ylab("log Bayes Factors") +
theme_classic() +
#theme(axis.text.y=element_blank(), axis.title.y=element_blank()) +
coord_flip()
```
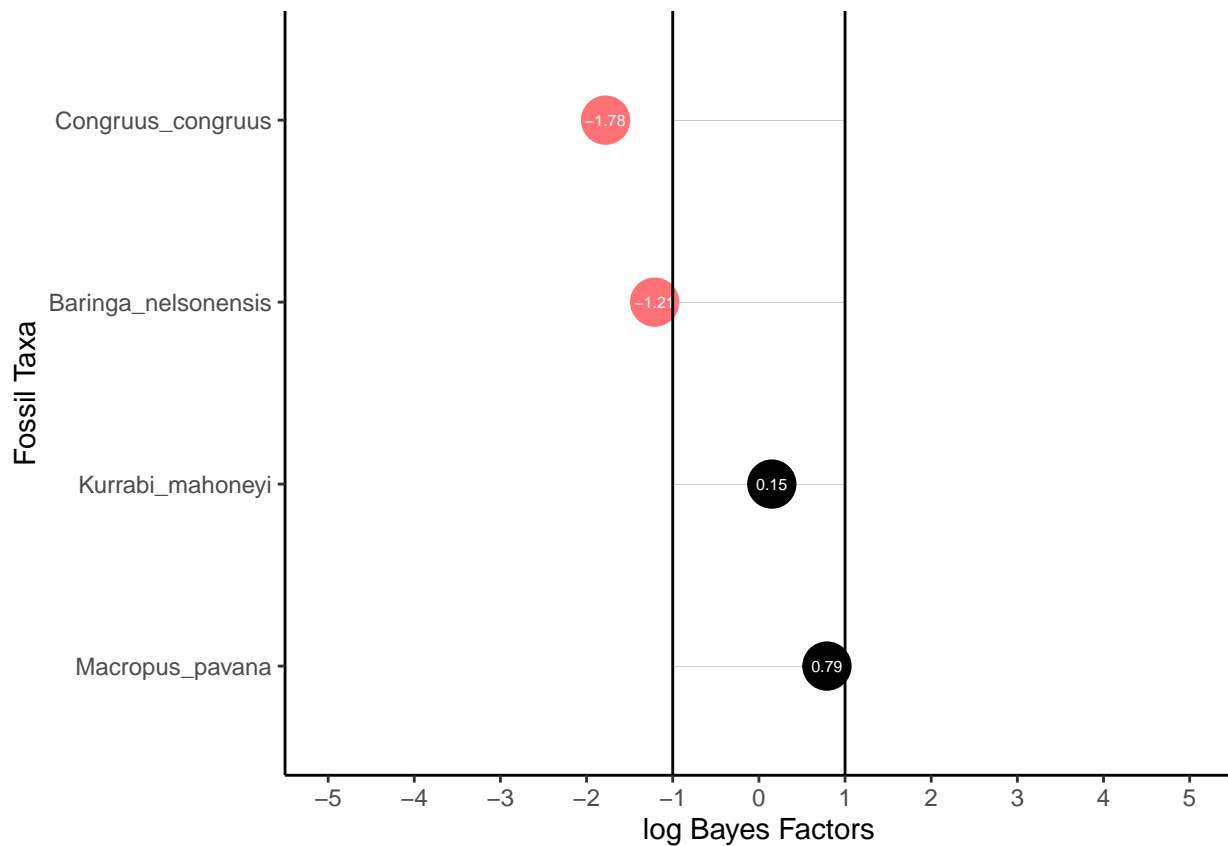

Investigating estimated ages of fossils

Create a function to pull the ages of each fossil taxon estimated

```
get.fossil.ages <- function(fossil.tips, trees){
  tree.tables <- lapply(trees, print.tree)
  fossil.tables <- lapply(1:length(tree.tables), function(x) {
    subset(tree.tables[[x]], tree.tables[[x]]$label %in% fossil.tips)
  })
```

```
  fossil.ages <- lapply(1:length(fossil.tables), function(x) {
    select(fossil.tables[[x]], label, time_bp)
  })
  final <- bind_rows(fossil.ages)
}
```

Need to create a function called "print.tree" that I borrowed some code from biogeoBEARS

Now pull out the info on those fossils

```
my.test <- get.fossil.ages(fossil.tips = fossil_taxa, trees = post.trees)

(ggplot(my.test, aes(x=time_bp, y=label, fill=..x..))
  + scale_fill_gradientn(colours=wes_palette("Zissou1"))
  + geom_density_ridges_gradient(scale=1.5)
  + scale_x_reverse()
  + theme_classic())
```

```
## Picking joint bandwidth of 0.6
```