Estimate Fossil LMA

## Data Prep

Loading require packages

require(BIEN)

## Loading required package: BIEN

## Loading required package: RPostgreSQL

## Loading required package: DBI

## Type vignette("BIEN") to get started

## The BIEN database has been updated to version 4.1 (as of 31-10-2018)! This version includes nearly twice as many occurrence records and improved validations.

require(plyr)

## Loading required package: plyr

require(tidyr)

## Loading required package: tidyr

require(dplyr)

## Loading required package: dplyr

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

require(mosaic)

## Loading required package: mosaic

## Loading required package: lattice

## Loading required package: ggformula

## Loading required package: ggplot2

## Registered S3 methods overwritten by 'ggplot2':  
## method from   
## [.quosures rlang  
## c.quosures rlang  
## print.quosures rlang

## Loading required package: ggstance

##   
## Attaching package: 'ggstance'

## The following objects are masked from 'package:ggplot2':  
##   
## geom\_errorbarh, GeomErrorbarh

##   
## New to ggformula? Try the tutorials:   
## learnr::run\_tutorial("introduction", package = "ggformula")  
## learnr::run\_tutorial("refining", package = "ggformula")

## Loading required package: mosaicData

## Loading required package: Matrix

##   
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':  
##   
## expand

## Registered S3 method overwritten by 'mosaic':  
## method from   
## fortify.SpatialPolygonsDataFrame ggplot2

##   
## The 'mosaic' package masks several functions from core packages in order to add   
## additional features. The original behavior of these functions should not be affected by this.  
##   
## Note: If you use the Matrix package, be sure to load it BEFORE loading mosaic.

##   
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':  
##   
## mean

## The following object is masked from 'package:ggplot2':  
##   
## stat

## The following objects are masked from 'package:dplyr':  
##   
## count, do, tally

## The following object is masked from 'package:plyr':  
##   
## count

## The following objects are masked from 'package:stats':  
##   
## binom.test, cor, cor.test, cov, fivenum, IQR, median,  
## prop.test, quantile, sd, t.test, var

## The following objects are masked from 'package:base':  
##   
## max, mean, min, prod, range, sample, sum

require(stringr)

## Loading required package: stringr

require(lme4)

## Loading required package: lme4

##   
## Attaching package: 'lme4'

## The following object is masked from 'package:mosaic':  
##   
## factorize

require(magrittr)

## Loading required package: magrittr

##   
## Attaching package: 'magrittr'

## The following object is masked from 'package:tidyr':  
##   
## extract

## Mixed-Effects Modeling

First, we need to create the fossil data for prediction.

florissant\_fossil\_int <- readRDS("./data/processed/04\_florissant\_fossil\_clean.rds")  
renova\_fossil\_int <- readRDS("./data/processed/04\_renova\_fossil\_clean.rds")  
bridgecreek\_fossil\_int <- readRDS("./data/processed/04\_bridgecreek\_fossil\_clean.rds")  
  
fossil\_comb <- rbind(florissant\_fossil\_int,renova\_fossil\_int,bridgecreek\_fossil\_int)  
fossil\_gen <- fossil\_comb$Genus  
fossil\_gen <- unique(fossil\_gen)  
fossil\_gen\_df <- as.data.frame(fossil\_gen)  
fossil\_tax <- BIEN\_taxonomy\_genus(fossil\_gen)  
fossil\_tax <- fossil\_tax[-c(1,7:9)]  
fossil\_tax <- unique(fossil\_tax)  
  
all\_fossil <- rbind(florissant\_fossil\_int, renova\_fossil\_int, bridgecreek\_fossil\_int)  
all\_fossil$binomial <- paste(all\_fossil$Genus, all\_fossil$species)  
all\_fossil$binomial <- str\_replace\_all(all\_fossil$binomial,"\\s+","\_")  
all\_fossil <- all\_fossil[-c(1,2)]  
  
 all\_fossil\_LMEpred <- all\_fossil %>%  
 separate(binomial,   
 c("scrubbed\_genus", "species"))

## Warning: Expected 2 pieces. Additional pieces discarded in 39 rows [10, 48,  
## 50, 96, 97, 98, 104, 108, 286, 287, 288, 289, 294, 318, 324, 325, 343, 364,  
## 377, 384, ...].

all\_fossil\_royer\_pred <- left\_join(all\_fossil\_LMEpred, fossil\_tax, by = "scrubbed\_genus")  
  
all\_fossil\_royer\_pred <- na.omit(all\_fossil\_royer\_pred)  
all\_fossil\_royer\_pred <- unique(all\_fossil\_royer\_pred)  
  
colnames(all\_fossil)[colnames(all\_fossil)=="Petiole Width (cm)"] <- "avg\_petiole\_width"  
colnames(all\_fossil)[colnames(all\_fossil)=="Leaf Area (cm^2)"] <- "avg\_LA"  
colnames(all\_fossil)[colnames(all\_fossil)=="PW^2/A"] <- "log\_pet\_leafarea"  
colnames(all\_fossil)[colnames(all\_fossil)=="LMA (g/m^2)"] <- "log\_LMA"

Next, we have to apply a fit linear mixed-effects model. By using the lmer function in the lme4 package, we are able to fit the model to the selected data.

royer\_tax\_full <- readRDS("./data/processed/07\_lm4\_royer")  
royer\_tax\_na\_omit<-na.omit(royer\_tax\_full)  
  
royer\_lme <- lmer(log\_lma~royer\_tax\_full$log\_pet\_leafarea + (1|order/scrubbed\_family/scrubbed\_genus), data=royer\_tax\_full)  
royer\_lme\_sum<- summary(royer\_lme)  
  
  
royer\_pred <- as.data.frame(predict(royer\_lme, newdata = all\_fossil\_royer\_pred, allow.new.levels= TRUE))

## Warning in pred + REvals: longer object length is not a multiple of shorter  
## object length

#lm predictions and models using family as factor  
royer\_fossil\_dropped <- all\_fossil\_royer\_pred %>%   
 filter(!grepl('Cercidiphyllaceae', scrubbed\_family)) %>%  
 filter(!grepl('Smilacaceae', scrubbed\_family)) %>%  
 filter(!grepl('Staphyleaceae', scrubbed\_family))  
  
royer\_fossil\_dropped <- all\_fossil\_royer\_pred %>%   
 filter(!grepl('Cercidiphyllaceae', scrubbed\_family)) %>%  
 filter(!grepl('Smilacaceae', scrubbed\_family)) %>%  
 filter(!grepl('Staphyleaceae', scrubbed\_family))  
  
royer\_tax\_fam\_count <- royer\_tax\_full%>%   
 group\_by(scrubbed\_family) %>%   
 tally() %>%  
 filter(!grepl('Unknown', scrubbed\_family))  
  
royer\_tax\_count <- left\_join(royer\_tax\_full, royer\_tax\_fam\_count, by="scrubbed\_family")

### Preparing Logistic Regression

Determining the top 5 extant families with the most samples, and the top 10 families respectively.

royer\_tax\_top5 <- subset(royer\_tax\_count, as.numeric(n)>23)  
royer\_tax\_over10 <- subset(royer\_tax\_count, as.numeric(n)>9)

Creating the linear models for comparing log lma and predictions for log(lma)

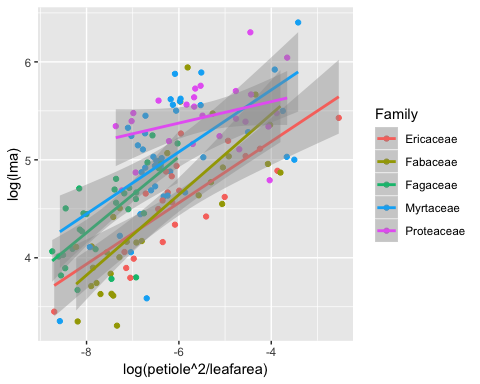
royer\_lmfam\_top5 <- lm(log\_lma~log\_pet\_leafarea, data = royer\_tax\_top5)  
pred\_lmfam\_top5 <- as.data.frame(predict(royer\_lmfam\_top5, interval="prediction"))

## Warning in predict.lm(royer\_lmfam\_top5, interval = "prediction"): predictions on current data refer to \_future\_ responses

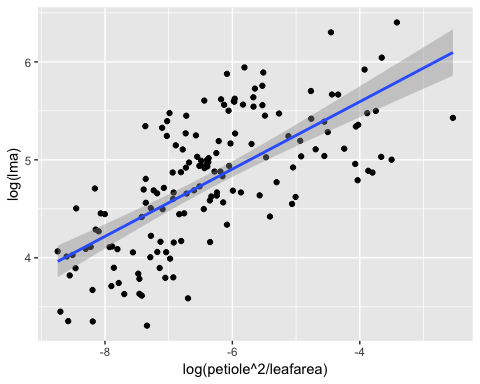
royer\_lmfam\_top5\_bound <- cbind(royer\_tax\_top5, pred\_lmfam\_top5)

Plotting log\_lma vs log(petiole^2/leafarea) for the top 5 families

ggplot(royer\_lmfam\_top5\_bound, aes(log\_pet\_leafarea, log\_lma))+  
 aes(color=royer\_lmfam\_top5\_bound$scrubbed\_family)+  
 geom\_point() +  
 geom\_smooth(method=lm, se=TRUE)+  
 labs(x="log(petiole^2/leafarea)")+  
 labs(y="log(lma)")+  
 labs(color = "Family")



ggplot(royer\_lmfam\_top5\_bound, aes(log\_pet\_leafarea, log\_lma))+  
 geom\_point() +  
 geom\_smooth(method=lm, se=TRUE)+  
 labs(x="log(petiole^2/leafarea)")+  
 labs(y="log(lma)")+  
 labs(color = "Family")



### Logistic Regression

Logistic regression predicting phenology off of log\_lma

###Create sets so we can determine the top 5 extant families with the most samples, and the top 10 families respecitively  
  
##Creation of fossil esitmates of top 5 fossil families  
fossil\_top5 <- royer\_fossil\_dropped%>%   
 group\_by( scrubbed\_family) %>%   
 tally()%>%  
 filter(!grepl('Unknown', scrubbed\_family))  
fossil\_top5 <- left\_join(fossil\_top5, royer\_fossil\_dropped, by="scrubbed\_family")  
fossil\_top5 <- subset(fossil\_top5, as.numeric(n)>22)  
  
#Creation of extant off of the five top families  
fossil\_top5\_tax <- as.data.frame(unique(fossil\_top5$scrubbed\_family))  
extant\_top5\_fossil\_tax <- subset(royer\_tax\_na\_omit, (royer\_tax\_na\_omit$scrubbed\_family %in% fossil\_top5\_tax))  
  
#creation of fossils that are in all extant families with more than 10 observations-----  
extant\_over10\_fossil\_tax <- subset(royer\_tax\_na\_omit, (royer\_tax\_na\_omit$scrubbed\_family %in% royer\_tax\_over10$scrubbed\_family))  
  
##Create royer\_data\_split so genus and species are separated  
royer\_data<-read.csv("./data/raw/royer\_data.csv")  
royer\_data\_split <-  
 royer\_data %>% separate(binomial, c("genus", "species"))

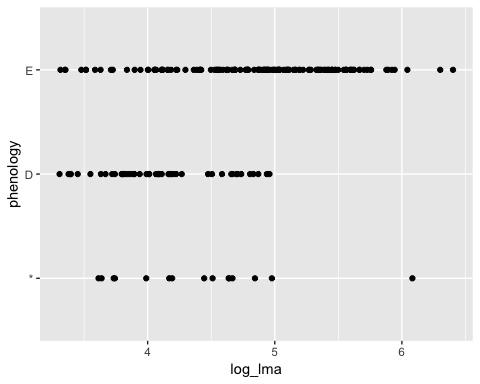
## Warning: Expected 2 pieces. Additional pieces discarded in 74 rows [11,  
## 15, 16, 20, 21, 22, 34, 35, 37, 38, 39, 40, 60, 61, 83, 84, 92, 117, 126,  
## 131, ...].

royer\_data\_split <- subset(royer\_data\_split, royer\_data\_split$genus %in% extant\_over10\_fossil\_tax$scrubbed\_genus)  
royer\_data\_split <- subset(royer\_data\_split, royer\_data\_split$species %in% extant\_over10\_fossil\_tax$species)

Logistic regression predictions on deciduous vs evergreen for fossils

royer\_data\_split <- unique(royer\_data\_split)  
colnames(royer\_data\_split)[colnames(royer\_data\_split)=="leaf\_area"] <- "avg\_LA"  
glm\_pred\_data <- left\_join(extant\_over10\_fossil\_tax, royer\_data\_split, by = "avg\_LA")  
glm\_pred\_data <- na.omit(glm\_pred\_data)  
glm\_pred\_data <- unique(glm\_pred\_data)  
  
  
ggplot(glm\_pred\_data, aes(log\_lma, phenology))+   
 geom\_point() +   
 stat\_smooth(method="glm", method.args=list(family="binomial"), se=FALSE)

## Warning: Computation failed in `stat\_smooth()`:  
## y values must be 0 <= y <= 1



ggplot(glm\_pred\_data, aes(log\_lma, phenology, group=(scrubbed\_family), color=scrubbed\_family)) +   
 geom\_point() +   
 stat\_smooth(method="glm", method.args=list(family="binomial"), se=FALSE)

## Warning: Computation failed in `stat\_smooth()`:  
## y values must be 0 <= y <= 1

