warmXtrophic Project: Plant Composition Data Analyses

Kara Dobson, Moriah Young

July 11, 2021

Load in packages & data

```r
# Clear all existing data
rm(list = ls())

# Load packages
library(tidyverse)
library(ggplot2)
library(lme4)
library(olsrr)
library(predictmeans)
library(car)
library(fitdistrplus)
library(ggpubr)
library(rstatix)
library(vegan)
library(interactions)
library(sjPlot)
library(effects)
library(glmmTMB)

# Set working directory
Sys.getenv("L1DIR")
```

```
## [1] "/Volumes/GoogleDrive/Shared drives/SpaCE_Lab_warmXtrophic/data/L1/"
```

```r
L1_dir <- Sys.getenv("L1DIR")
L2_dir <- Sys.getenv("L2DIR")
list.files(L1_dir)
```

```
##  [1] "ANPP"              "climate_data"      "CN"
##  [4] "greenup"           "herbivory"         "HOBO_data"
##  [7] "PAR"               "phenology"         "plant_composition"
## [10] "SLA"
```

```r
# read in plant comp data
comp <- read.csv(file.path(L1_dir, "plant_composition/final_plantcomp_L1.csv"))
comp <- comp %>% select(-X)  # get rid of 'X' column that shows up
```

```r
# adding sequential year variable starting at 1: this is because 2016... are
# large numbers compare with other values in the dataset. We can always label
# axes with these real years.
comp$year_factor[comp$year == 2015] <- 1
comp$year_factor[comp$year == 2016] <- 1
comp$year_factor[comp$year == 2017] <- 2
comp$year_factor[comp$year == 2018] <- 3
comp$year_factor[comp$year == 2019] <- 4
comp$year_factor[comp$year == 2020] <- 5

# Remove non-plant data - MY: not sure if I should remove these?
comp <- comp[!(comp$species == "Bare_Ground" | comp$species == "Unknown" | comp$species ==
    "Brown" | comp$species == "Litter" | comp$species == "Vert_Litter" | comp$species ==
    "Animal_Disturbance"), ]
```

```r
# calculating total composition sums - proxy for most common species
comp_yearly_totals <- comp %>% group_by(species, site, year, plot) %>% summarize(comp_sum = sum(cover,
    na.rm = T))
```

```
## ‘summarise()‘ has grouped output by ’species’, ’site’, ’year’. You can override using the ‘.groups‘ a
```

Relative abundance

```r
# Calculating relative abundance
comp1 <- comp %>% select(species, site, year, plot, cover, state)
plot_mean <- aggregate(cover ~ plot * species * year * site * state, data = comp1,
    FUN = mean, na.rm = T)
names(plot_mean)[names(plot_mean) == "cover"] <- "plot_mean"  # change ’cover’ column name to ’plot_mea
View(plot_mean)

# convert cover to relative abundance first get summed cover for all plants per
# plot
plot_cover_total = aggregate(plot_mean ~ plot * year * site * state, data = plot_mean,
    FUN = sum, na.rm = T)
names(plot_cover_total)[names(plot_cover_total) == "plot_mean"] <- "plot_cover_total"
View(plot_cover_total)
comp2 <- merge(plot_mean, plot_cover_total, by = c("plot", "year", "site", "state"))

# calculate relative percent cover per species in each quadrat (=’relative
# abundance’)
comp2$relabun <- comp2$plot_mean/comp2$plot_cover_total
summary(comp2)
```

```
##      plot                year           site               state          
##  Length:2472        Min.   :2015   Length:2472        Length:2472       
##  Class :character   1st Qu.:2016   Class :character   Class :character  
##  Mode  :character   Median :2017   Mode  :character   Mode  :character  
##                     Mean   :2017                                        
##                     3rd Qu.:2019                                        
##                     Max.   :2020                                        
##    species            plot_mean       plot_cover_total    relabun        
##  Length:2472        Min.   :  0.500   Min.   : 15.88   Min.   :0.005242  
##  Class :character   1st Qu.:  1.816   1st Qu.: 57.75   1st Qu.:0.023009
##  Mode  :character   Median :  4.445   Median : 85.83   Median :0.054084
##                     Mean   :  9.515   Mean   : 90.90   Mean   :0.116505
##                     3rd Qu.: 11.500   3rd Qu.:124.18   3rd Qu.:0.148558
##                     Max.   :110.000   Max.   :190.75   Max.   :0.809170
```

```
# create dataframes for kbs and umbs - remember that these contain species within
# plots
comp_kbs <- subset(comp2, site == "kbs")
comp_umbs <- subset(comp2, site == "umbs")
```

Relative Abundance for comparisons between native and exotic species

```
# getting relative % cover for comparisons between native & exotic most code from
# Kileigh's old script average sub-quadrats for plots - MY: what are sub
# quadrats? Just plots?
comp_org <- subset(comp, origin == "Exotic" | origin == "Native")
plot_mean_org <- aggregate(cover ~ plot * origin * species * year * site * state,
    data = comp_org, FUN = mean, na.rm = T)
names(plot_mean_org)[names(plot_mean_org) == "cover"] <- "plot_mean_org"
head(plot_mean_org)
```

```
##   plot origin species year site   state plot_mean_org
## 1   A1 Native    Acmi 2015  kbs ambient     13.625000
## 2   A3 Native    Acmi 2015  kbs ambient      9.111111
## 3   A6 Native    Acmi 2015  kbs ambient      6.333333
## 4   B1 Native    Acmi 2015  kbs ambient      6.777778
## 5   B4 Native    Acmi 2015  kbs ambient     15.444444
## 6   B5 Native    Acmi 2015  kbs ambient      6.250000
```

```
# convert cover to relative abundance first get summed cover for all plants per
# plot
cover_sum = aggregate(plot_mean_org ~ plot * origin * year * site * state, data = plot_mean_org,
    FUN = sum, na.rm = T)
names(cover_sum)[names(cover_sum) == "plot_mean_org"] <- "cover_sum"
head(cover_sum)
```

```
##   plot origin year site   state cover_sum
## 1   A1 Exotic 2015  kbs ambient  22.38333
## 2   A3 Exotic 2015  kbs ambient  44.24762
## 3   A6 Exotic 2015  kbs ambient  24.02222
## 4   B1 Exotic 2015  kbs ambient  13.00000
## 5   B4 Exotic 2015  kbs ambient  34.08730
## 6   B5 Exotic 2015  kbs ambient  55.80000
```

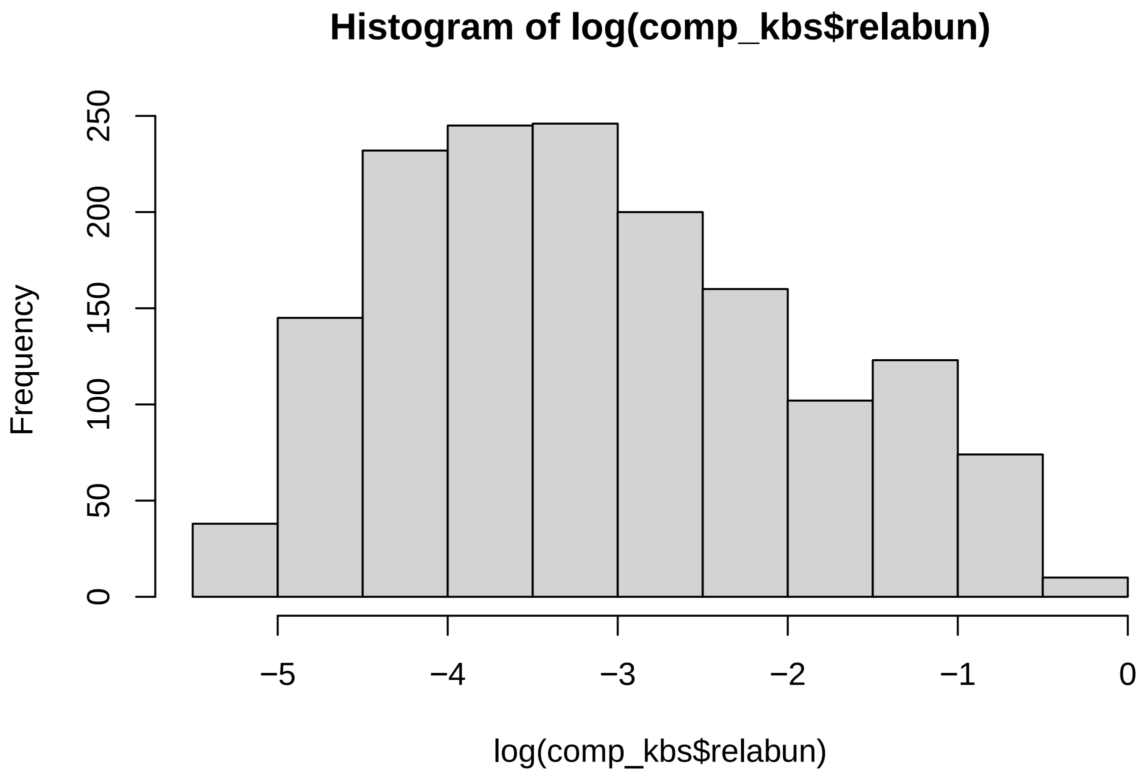
```
comp2 <- merge(plot_mean_org, cover_sum, by = c("plot", "origin", "year", "site",
    "state"))

# calculate relative percent cover per species in each quadrat (='relative
# abundance')
comp2$relabun <- comp2$plot_mean_org/comp2$cover_sum
summary(comp2)
```

```
##      plot              origin               year          site          
##  Length:1914        Length:1914        Min.   :2015   Length:1914       
##  Class :character   Class :character   1st Qu.:2016   Class :character  
##  Mode  :character   Mode  :character   Median :2017   Mode  :character  
##                                        Mean   :2017                     
##                                        3rd Qu.:2019                     
##                                        Max.   :2020                     
##     state             species          plot_mean_org       cover_sum     
##  Length:1914        Length:1914        Min.   :  0.500   Min.   :  1.00  
##  Class :character   Class :character   1st Qu.:  1.667   1st Qu.: 19.35  
##  Mode  :character   Mode  :character   Median :  4.333   Median : 30.62  
##                                        Mean   :  9.553   Mean   : 33.95  
##                                        3rd Qu.: 11.000   3rd Qu.: 45.46  
##                                        Max.   :110.000   Max.   :126.00  
##     relabun        
##  Min.   :0.009009  
##  1st Qu.:0.065864  
##  Median :0.160139  
##  Mean   :0.299373  
##  3rd Qu.:0.457948  
##  Max.   :1.000000  
```

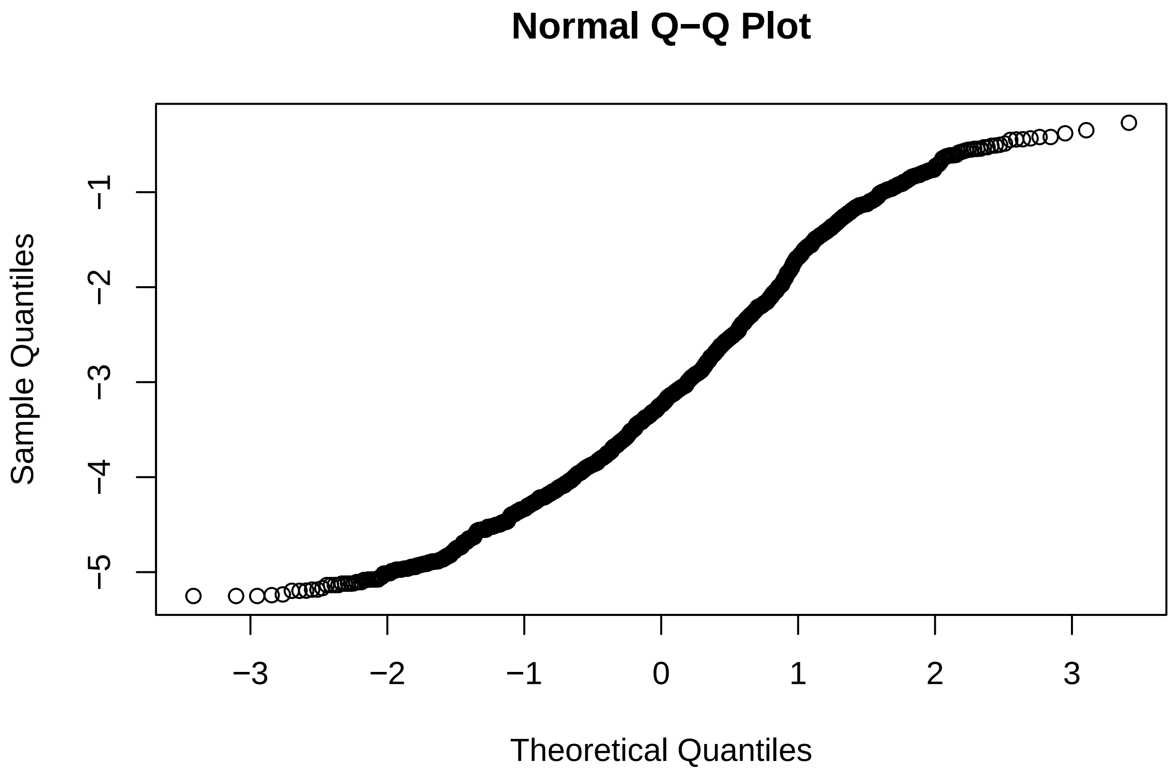
```r
# create dataframes for kbs and umbs - remember that these contain species within
# plots
comp_org_kbs <- subset(comp2, site == "kbs")
comp_org_umbs <- subset(comp2, site == "umbs")
```

```r
# https://rfunctions.blogspot.com/2016/08/comparing-and-estimating-species.html
```

```r
# checking for normality in raw data
hist(log(comp_kbs$relabun))
```



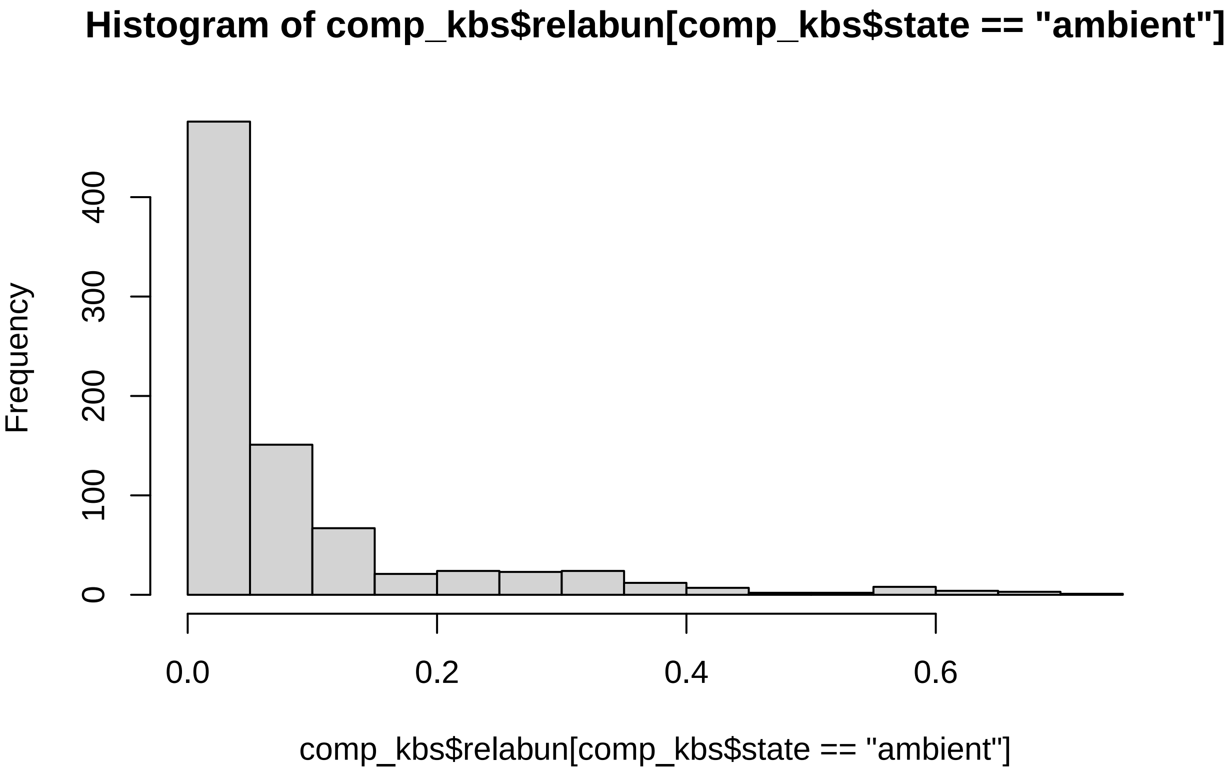
```
qqnorm(log(comp_kbs$relabun))
```



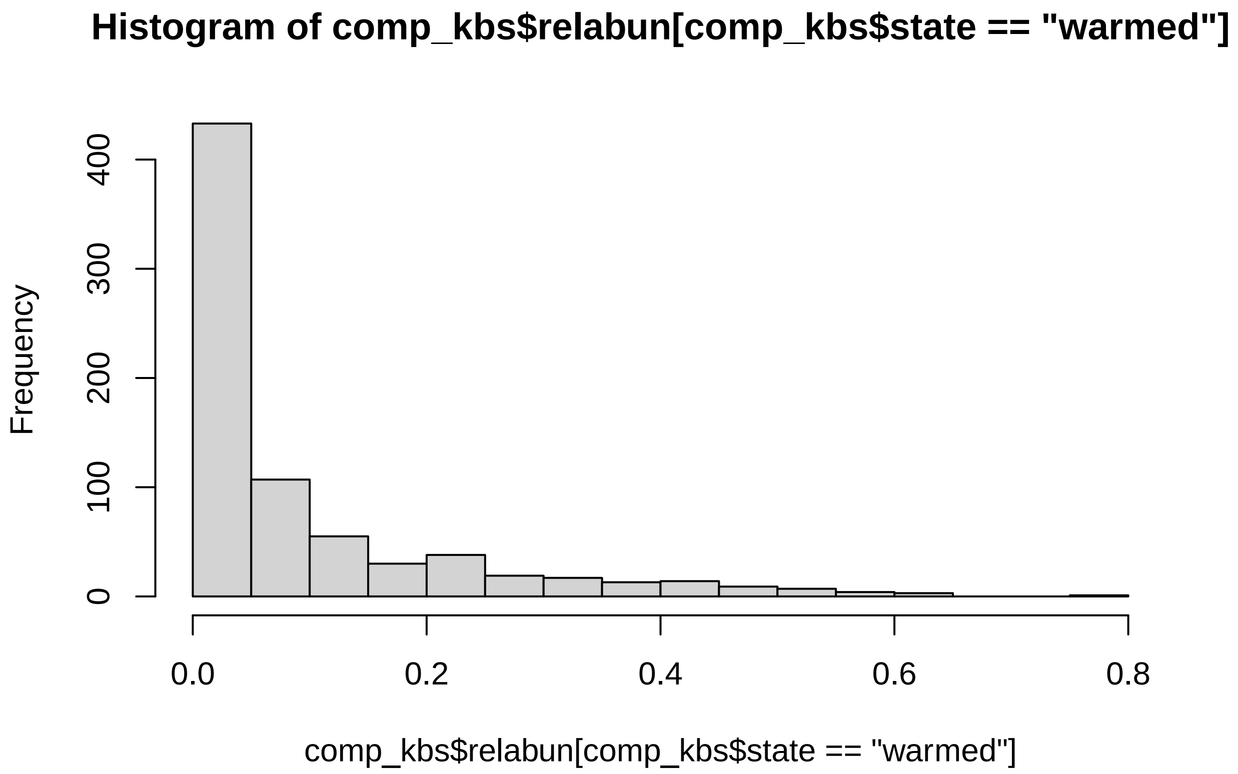
```r
shapiro.test(log(comp_kbs$relabun))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  log(comp_kbs$relabun)
## W = 0.9707, p-value < 2.2e-16
```

```r
hist(comp_kbs$relabun[comp_kbs$state == "ambient"])
```



```r
hist(comp_kbs$relabun[comp_kbs$state == "warmed"])
```


Histogram of comp_kbs$relabun[comp_kbs$state == "warmed"]
Frequency
0
100
200
300
400
0.0
0.2
0.4
0.6
0.8
comp_kbs$relabun[comp_kbs$state == "warmed"]