Solanum Analysis and Visualisation

Christopher Dudley

Packages

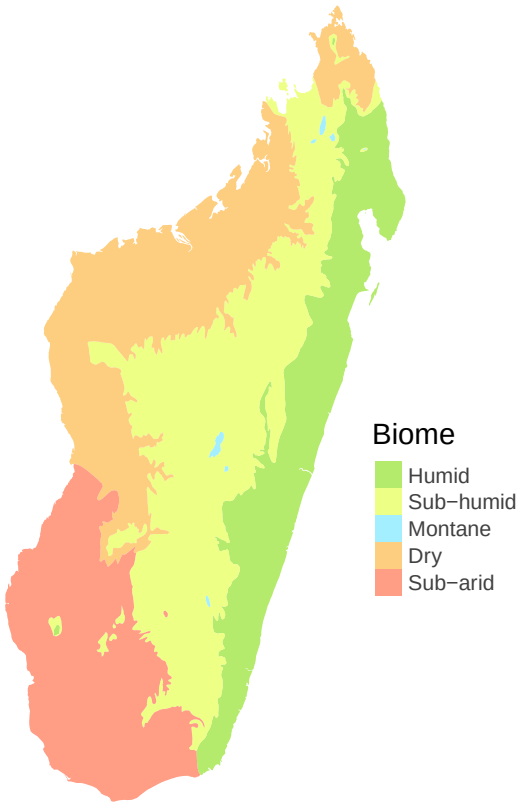
```
library(tidyverse) # for general data manipulation and visualisation
library(sf) # for reading in shapefiles and geospatial analysis
library(raster) # for reading in raster files and geospatial analysis
library(tmap) # for cartography
library(FactoMineR) # for PCA
library(factoextra) # for PCA
library(ggcorrplot) # for correlation plots
library(cowplot) # for multi-plot
library(ape) # for reading and manipulating tree files
library(ggtree) # for viewing trees
```

Basic Map with Biomes

Madagascar can be broken into five distinct biomes and it is good to get a visualisation of these biomes. We can create a simple, yet effective, map using the `tmap` package. There are other options for creating maps, such as `leaflet` for interactive maps or `ggplot`, but `tmap` is the best when it comes to efficient, attractive maps.

```
# biome data, converted to sf object
(biomes <- st_read('bioclimates/', layer = 'bc5_dd'))
(biomes <- biomes %>%
  mutate(BC1 = as.factor(BC1)) %>%
  st_set_crs(value = 4326) %>% # WGS84 CRS
  arrange(BC1))

# madagascar biomes base map
pal <- c('#B5EB6C', '#EEFF85', '#A4EFFF', '#FECE80', '#FF9D85')

tmBaseMap <- tm_shape(biomes) +
  tm_polygons(col = 'BC1',
              title = 'Biome',
              labels = c('Humid', 'Sub-humid', 'Montane', 'Dry', 'Sub-arid'),
              border.alpha = 0,
              palette = pal)

# biome map figure (extra figure for presentations)
# (tmBiomeMap <- tmBaseMap  +
#  tm_layout(frame = FALSE,
#            legend.position = c(0.89, 0.26),
#            legend.width = 1,
#            legend.text.color = '#404040',
#            legend.title.fontfamily = 'Helvetica',
#            legend.text.fontfamily = 'Helvetica'))
# tmap_save(tmBiomeMap, file.path('..', 'results', 'figures', 'baseMap.pdf'))
```

```
# hypothesis map figure (extra figure for presentations)
# (tmHypMap <- tmBaseMap  +
#  tm_layout(frame = FALSE,
#            legend.position = c(0.99, 0.4),
#            legend.width = 1,
#            legend.text.color = '#404040',
#            legend.title.fontfamily = 'Helvetica',
#            legend.text.fontfamily = 'Helvetica'))
# tmap_save(tmBiomeMap, file.path('..', 'results', 'figures', 'hypMap.pdf'))
```



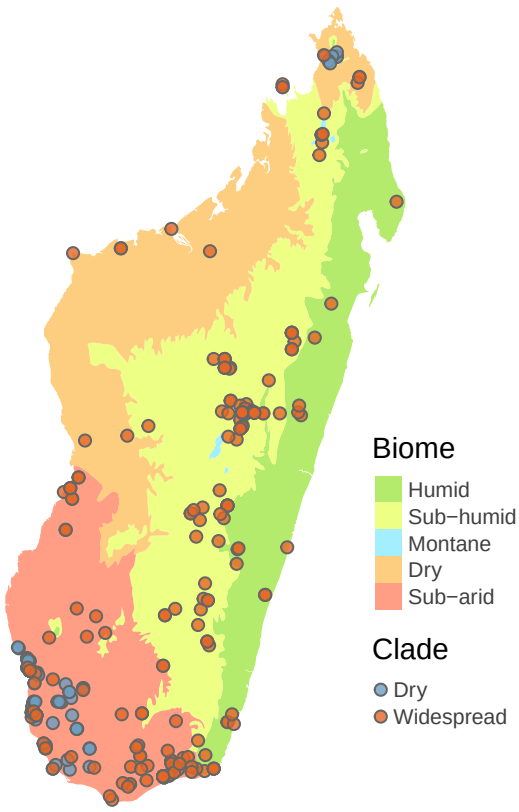
Distribution Map

We want to create a map showing the distribution of our two sub-clades and so we need to load in the distribution data, clean it and convert it to an `sf` object so that the points can be plotted using `tmap`.

We are interested in more than just plotting the points; we also want to know which biome each specimen is located in. A map can visualise this, but if we want to use that data for analysis, it needs to be extracted from the data. The `st_intersects` function from the `sf` package takes two spatial objects and returns a list of the intersections between the two objects. In our case, the `sol` database contains spatial points and the `biomes` data contains spatial polygons, so our results give us the polygon which our points fall in. As we know which polygons relate to which biome, we simply use `dplyr` functions to add these results to the `sol` data to be used for future analysis.

```
# distribution data, converted to sf object
(sol <- read_csv('sol.csv'))
(sol <- sol %>%
  rename(brahms = BRAHMS,
         species = SPECIES,
```

```
         clade = GROUP,
         lat = LATDEC,
         lng = LONGDEC) %>%
  mutate(brahms = as.factor(brahms),
         clade = as.factor(clade),
         species = as.factor(str_extract(species, ('\\w*\\s\\w*')))) %>%
  st_as_sf(coords = c('lng', 'lat'), crs = 4326))

# determine which biome a species falls in
(inter <- as_tibble(unlist(st_intersects(sol, biomes, sparse = FALSE))))

# join the species/biome data with the distribution data to filter out species
# which don't intersect with a biome
(sol <- inter %>%
  add_column(brahms = sol$brahms, .before = 1) %>%
  gather(biome, value, -brahms) %>%
  filter(value == TRUE) %>%
  dplyr::select(brahms, biome) %>%
  mutate(biome = as.integer(str_extract(biome, '\\d+')),
         biome = case_when(biome <= 5 ~ 1,
                           between(biome, 6, 17) ~ 2,
                           between(biome, 18, 24) ~ 3,
                           between(biome, 25, 29) ~ 4,
                           biome >= 30 ~ 5)) %>%
  inner_join(sol, by = 'brahms') %>%
  dplyr::select(brahms, species, clade, biome, geometry) %>%
  arrange(clade, species, biome) %>%
  st_as_sf())

# tmap distribution map
tmDistMap <- tmBaseMap +
  tm_shape(sol) +
  tm_bubbles(shape = 21, size = 0.1, alpha = 0.8, col = 'clade',
             title.col = 'Clade',
             labels = c('Dry', 'Widespread'),
             palette = c('#6D9EC1', '#E46726')) +
  tm_layout(frame = FALSE,
            legend.position = c(0.89, 0.1),
            legend.width = 1,
            legend.text.color = '#404040',
            legend.title.fontfamily = 'Helvetica',
            legend.text.fontfamily = 'Helvetica')
tmap_save(tmDistMap, file.path('..', 'results', 'figures', 'distMap.pdf'))
```



Species/Biome Interaction

To create an ancestral state reconstruction to find the origin biome of our Malagasy clade, we first need to know the current biomes of the species within the clade. Earlier, we found the biome that each species was found in using the `st_intersects` function from the `sf` package and added this information to the `sol` data. We can now simply use the `dplyr` functions to summarise how often each species is found in a particular biome. For our purposes, we considered a species to exist in a biome if it was found more than 12.5% of the time. We chose this number because one of our species only had 8 samples and 1 sample out of 8 is 12.5%. This helps to make sure that we aren't considering a species to exist in a biome if it only has a minor presence. In this way, incorrect identifications, species on ecotones or poor GPS data won't provide false results.

```
# get occurences of species in biome; if > 12.5% found in biome, species is
# considered to be found in that biome
(sol %>%

  # remove geometry column and convert to tibble
  st_set_geometry(NULL) %>%
  as_tibble() %>%

  # count the number of each species found in each biome
  group_by(clade, biome) %>%
  count(species) %>%
  ungroup() %>%
  arrange(clade, species, biome) %>%

  # turn counts into percentages of species occurence in each biome
  group_by(species) %>%
```

```
  mutate(count = sum(n),
         n = round(n / sum(n), digits = 3)) %>%
  spread(biome, n) %>%
  replace(is.na(.), 0) %>%
  rename('humid' = '1',
         'sub-humid' = '2',
         'montane' = '3',
         'dry' = '4',
         'sub-arid' = '5'))
```

clade	species	count	humid	sub-humid	montane	dry	sub-arid
1	Solanum bumeliifolium	22	0.000	0.000	0.000	0.000	1.000
1	Solanum heinianum	36	0.000	0.000	0.000	0.000	1.000
1	Solanum mahoriense	8	0.000	0.250	0.000	0.750	0.000
1	Solanum toliaraea	17	0.000	0.000	0.000	0.000	1.000
2	Solanum batoides	33	0.030	0.030	0.000	0.000	0.939
2	Solanum croatii	28	0.036	0.000	0.000	0.000	0.964
2	Solanum erythracanthum	99	0.172	0.505	0.000	0.111	0.212
2	Solanum myoxotrichum	35	0.229	0.629	0.143	0.000	0.000
2	Solanum pyracanthos	22	0.136	0.045	0.000	0.000	0.818

BioClim Data

We wanted the climate data for each specimen in our database, so we used the `extract` function from the `raster` package to match the WorldClim data with the coordinates of the species in the `sol` data. We put these values in a new tibble called `bioClim`.

```
# worldclim data at 30s resolution; from http://worldclim.org/current
climRasters <- list.files('bio_37/', pattern = '.bil', full.names = TRUE)
climRasters <- map(climRasters, raster) # provides a list of single-layer rasters

# raster layer names; from http://worldclim.org/bioclim
varNames <- c('annual_Mean_Temperature', 'mean_Diurnal_Range', 'isothermality',
              'temperature_Seasonality', 'max_Temperature_of_Warmest_Month',
              'min_Temperature_of_Coldest_Month', 'temperature_Annual_Range',
              'mean_Temperature_of_Wettest_Quarter',
              'mean_Temperature_of_Driest_Quarter',
              'mean_Temperature_of_Warmest_Quarter',
              'mean_Temperature_of_Coldest_Quarter', 'annual_Precipitation',
              'precipitation_of_Wettest_Month', 'precipitation_of_Driest_Month',
              'precipitation_Seasonality', 'precipitation_of_Wettest_Quarter',
              'precipitation_of_Driest_Quarter', 'precipitation_of_Warmest_Quarter',
              'precipitation_of_Coldest_Quarter')

# extract bioClim data from distribution data
bioClim <- map_dfc(climRasters, function(layer)
                                  raster::extract(layer, as_Spatial(sol)@coords,
                                                  cellnumbers = FALSE))

# add brahms number and species columns; rename bioClim columns
bioClim <- bioClim %>%
```

```
  add_column(brahms = sol$brahms,
             species = sol$species,
             clade = sol$clade,
             .before = 1) %>%
  rename_at(vars(V1:V19), ~ varNames) %>%
  na.omit()
```

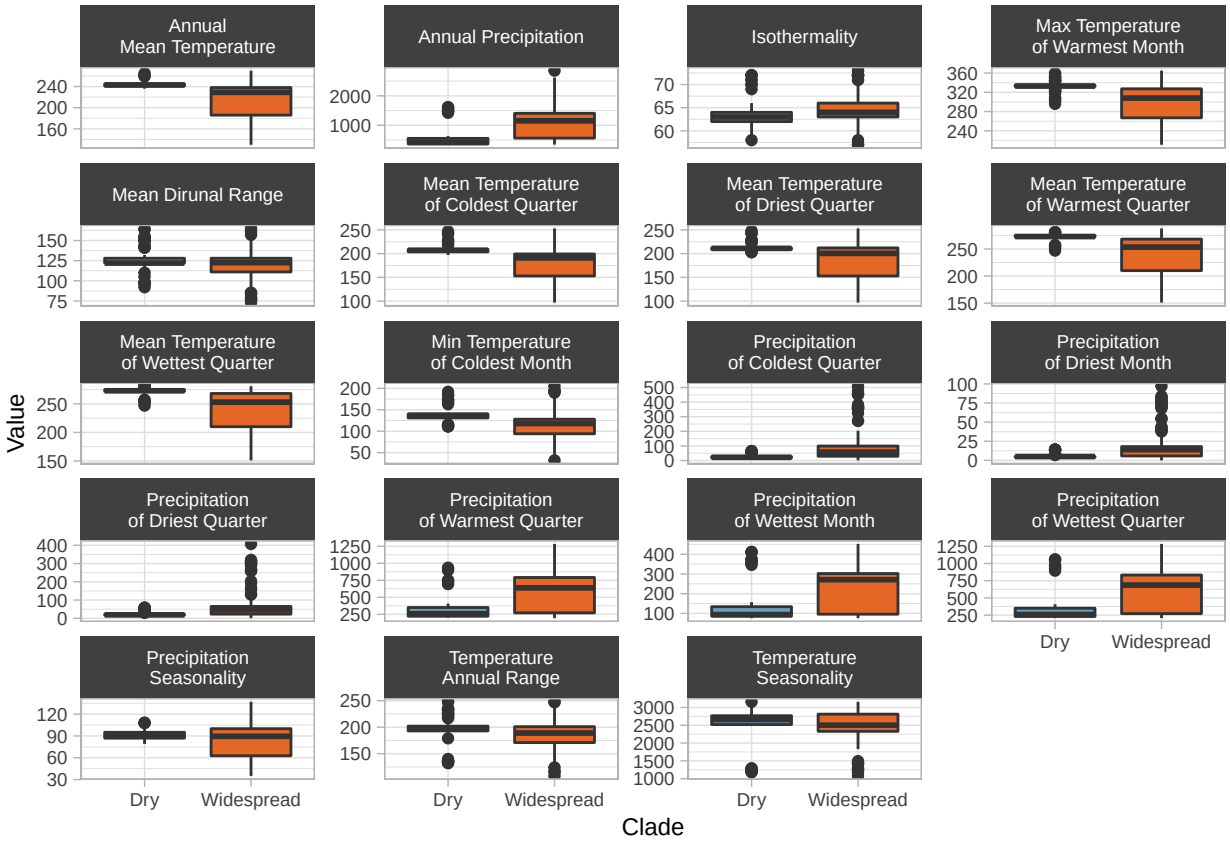
Stats

We can use some basic statistical analyses to visualise and understand the difference between the two sub-clades in terms of climate. Analyses of variance can show that there are significant difference between the two sub-clades when it comes to climate. Using boxplots is a fantastic way to visualise this difference and really helps to show that the widespread clade is truly widespread, even encompassing the climate measures of the dry clade.

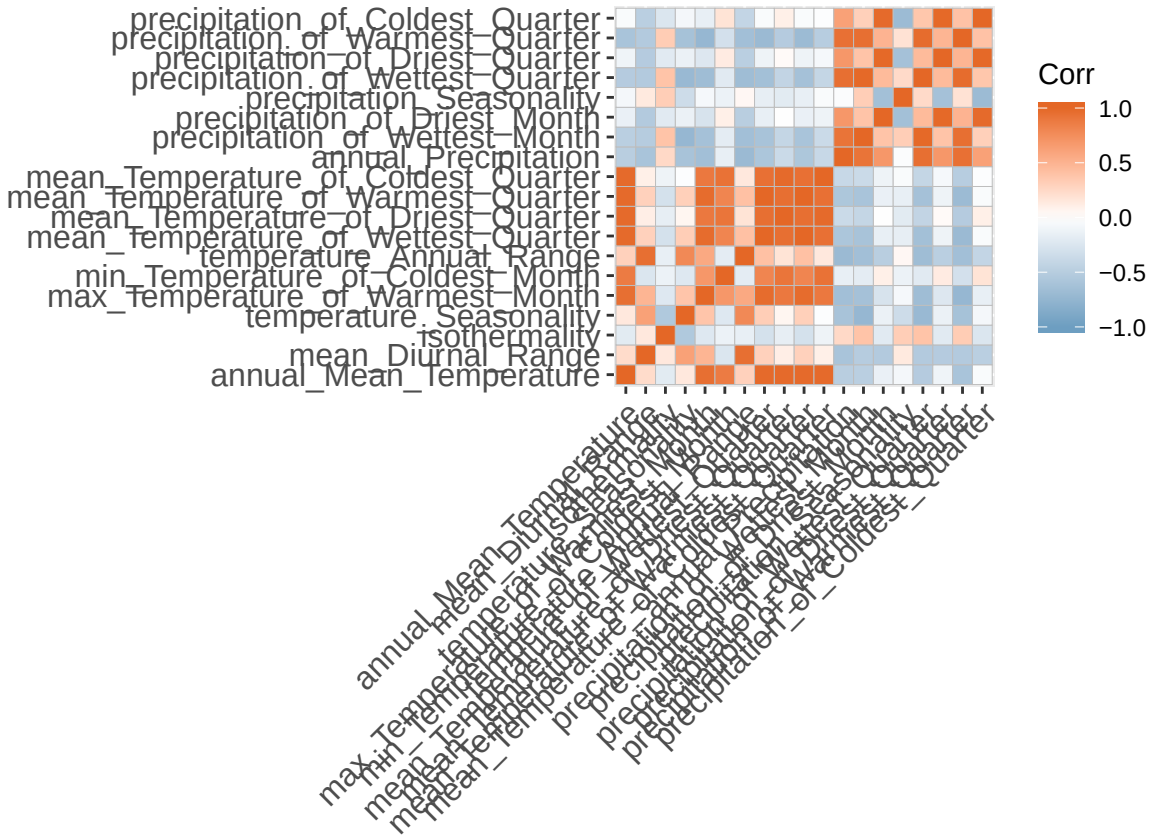
A quick glance at the WorldClim variables shows that many of them are probably going to be highly correlated. We can use the `ggcorrplot` function from the `ggcorrplot` package to verify our suspicions. This isn't an issue, but we should perform a PCA to reduce the dimensionality and look at the most influential variables for determining the difference between the two sub-clades.

```
# anova; 17 of 19 significant; only isothermality and temperature_Seasonality not
summary(aov(as.matrix(cbind(bioClim[4:22])) ~ bioClim$clade))

# boxplots
(plotNames <- varNames %>%
  str_replace_all('_', ' ') %>%
  str_to_title())

## alter bioclim variable names for plot; unfortunately need to do this manually
## because of how names get cut off in the plot
plotNames <- c('annual_Mean_Temperature' = 'Annual\nMean Temperature',
               'mean_Diurnal_Range' = 'Mean Dirunal Range',
               'isothermality' = 'Isothermality',
               'temperature_Seasonality' = 'Temperature\nSeasonality',
               'max_Temperature_of_Warmest_Month' = 'Max Temperature\nof Warmest Month',
               'min_Temperature_of_Coldest_Month' = 'Min Temperature\nof Coldest Month',
               'temperature_Annual_Range' = 'Temperature\nAnnual Range',
               'mean_Temperature_of_Wettest_Quarter' =
                                    'Mean Temperature\nof Wettest Quarter',
               'mean_Temperature_of_Driest_Quarter' =
                                    'Mean Temperature\nof Driest Quarter',
               'mean_Temperature_of_Warmest_Quarter' =
                                    'Mean Temperature\nof Warmest Quarter',
               'mean_Temperature_of_Coldest_Quarter' =
                                    'Mean Temperature\nof Coldest Quarter',
               'annual_Precipitation' = 'Annual Precipitation',
               'precipitation_of_Wettest_Month' = 'Precipitation\nof Wettest Month',
               'precipitation_of_Driest_Month' = 'Precipitation\nof Driest Month',
               'precipitation_Seasonality' = 'Precipitation\nSeasonality',
               'precipitation_of_Wettest_Quarter' = 'Precipitation\nof Wettest Quarter',
               'precipitation_of_Driest_Quarter' = 'Precipitation\nof Driest Quarter',
               'precipitation_of_Warmest_Quarter' = 'Precipitation\nof Warmest Quarter',
               'precipitation_of_Coldest_Quarter' = 'Precipitation\nof Coldest Quarter')
```

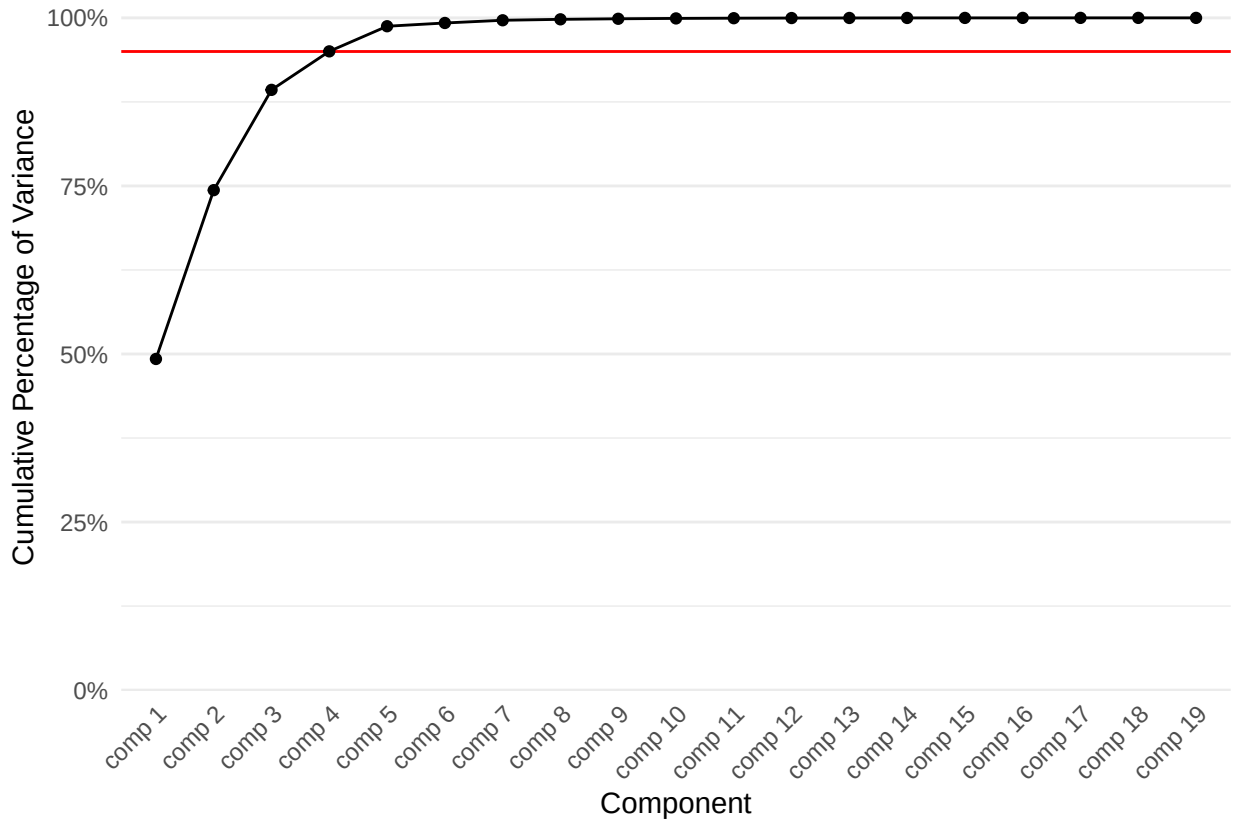
```
(boxplots <- bioClim %>%
  gather(key = "measurement", value = "value", -c(brahms, species, clade)) %>%
  na.omit() %>%
  ggplot(aes(x = clade, y = value, fill = clade)) +
    geom_boxplot() +
    scale_x_discrete(labels = c('Dry', 'Widespread')) +
    scale_fill_manual(values = c('#6D9EC1', '#E46726')) +
    xlab('Clade') +
    ylab('Value') +
    facet_wrap(~measurement, scales = 'free_y', nrow = 5, ncol = 4,
               labeller = as_labeller(plotNames)) +
    theme_light() +
    theme(legend.position = 'None',
          text = element_text(size = 9),
          strip.background = element_rect(fill = '#404040'),
          strip.text = element_text(family = ('Helvetica')),
          axis.text = element_text(colour = '#404040', family = 'Helvetica')))
```



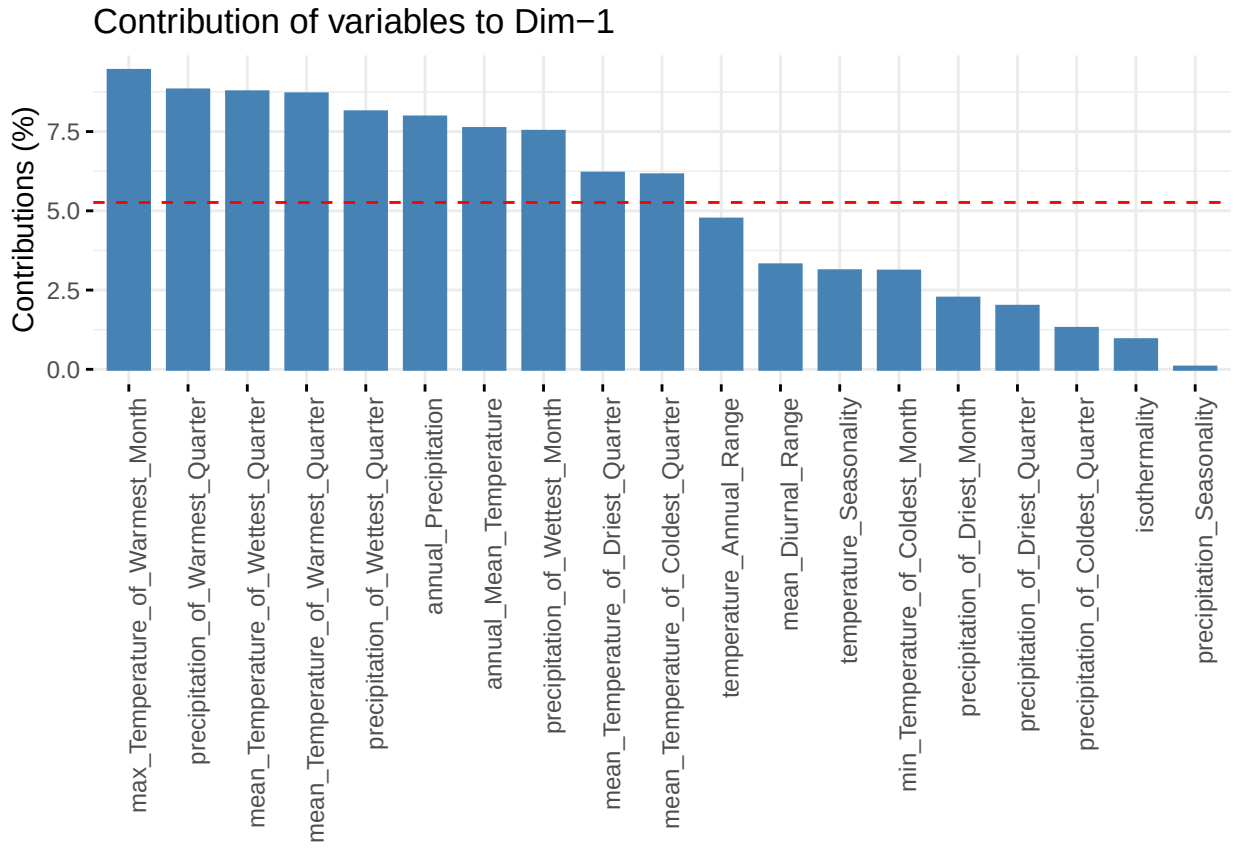

```
# ggsave(plot = boxplots, file.path('..', 'results', 'figures', 'boxplot.pdf'))

# view correlations between climate variables
(bioClim %>%
  dplyr::select(-brahms, -species, -clade) %>%
  cor() %>%
  ggcorrplot(ggtheme = ggplot2::theme_gray,
             colors = c('#6D9EC1', 'white', '#E46726')))
```




```r
# pca
climPCA <- PCA(bioClim, quali.sup = 1:3, graph = FALSE)
summary(climPCA)

# cumulative percentage of variance graph
climPCA$eig %>%
  as_tibble(rownames = 'component') %>%
  mutate(component = as.factor(component),
         component = fct_inorder(component),
         `cumulative percentage of variance` = `cumulative percentage of variance` / 100) %>%
  ggplot(aes(x = component, y = `cumulative percentage of variance`, group = 1)) +
    geom_hline(yintercept = 0.95, colour = 'red') +
    geom_point() +
    geom_line() +
    scale_y_continuous(labels = scales::percent,
                       limits = c(0, 1),
                       expand = expand_scale(mult = c(0, 0.05))) +
    labs(x = 'Component',
         y = 'Cumulative Percentage of Variance') +
    theme_minimal() +
    theme(panel.grid.major.x = element_blank(),
          axis.text.x = element_text(angle = 45,
                                     hjust = 1))
```




```
# dimdesc(climPCA, axes = 1:5) # not working with quali.sup arg in PCA

climPCA$var$contrib %>%
  as_tibble(rownames = 'variable') %>%
  arrange(desc(Dim.1))

contribPlot <- function(x) {fviz_contrib(climPCA, choice = 'var',
                                         axes = x, xtickslab.rt = 90)}
contribPlot(1) # contribution plots for the first principal component
```

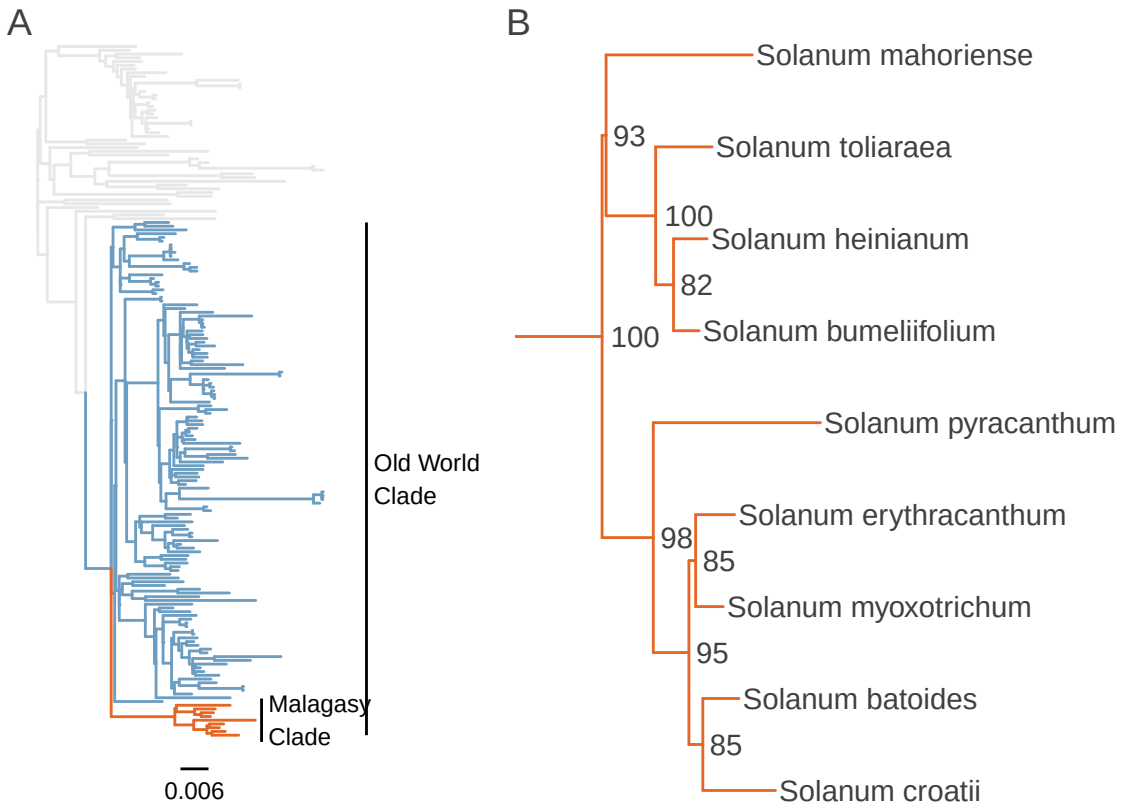


Phylogeny

After receiving the sequence results from the lab, we visualise the tree using the ggtree package. `ape` is used to read in the tree file and the Old-World and Malagasy clades are highlighted using the `MRCA` (most recent common ancestor) function. The `plot_grid` function from the `cowplot` package allows us to plot both the full tree and the clade tree next to each other.

```
# load in tree and provide node and tip labels
tree <- ape::read.nexus(file.path('..', 'results', 'Final_tree'))
tree$node.label <- c(100, 93, 100, 82, 98, 95, 85, 85)
tree$tip.label <- str_replace_all(tree$tip.label,'_', ' ')

# group tree according to clades
malagasy_clade <- MRCA(tree, c('Solanum batoides', 'Solanum mahoriense'))
oldworld_clade <- MRCA(tree, c('Solanum batoides', 'XAS119'))
tree <- groupClade(tree, .node = c(oldworld_clade, malagasy_clade), group_name = 'group')

# figure
palette <- c('#E7E7E7', '#6D9EC1', '#E46726')

# full tree
p <- ggtree(tree, ladderize = FALSE, aes(color = group)) +
       geom_treescale(y = -8, offset = -8, fontsize = 3) +
       scale_colour_manual(values = palette) +
       geom_cladelabel(node = malagasy_clade, label = 'Malagasy\nClade',
                       offset = 0.0004, offset.text = 0.001,
```

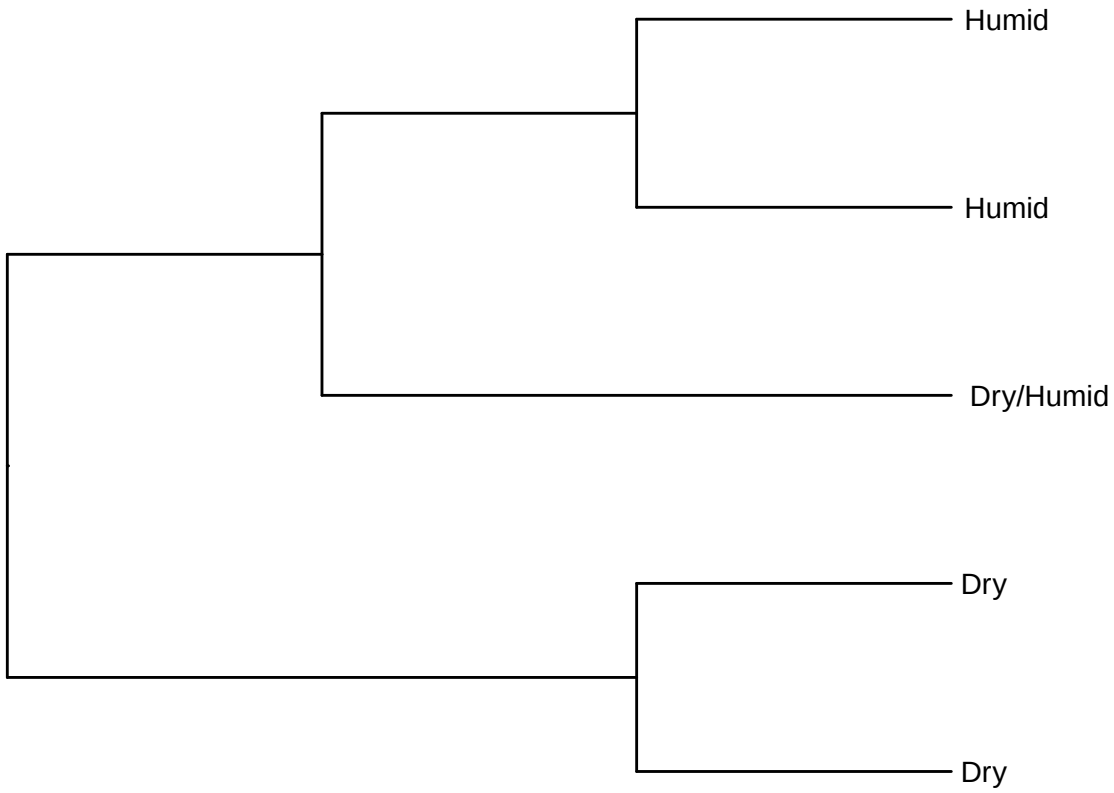
```
                        extend = 1.8, fontsize = 3) +
       geom_cladelabel(node = oldworld_clade, label = 'Old World\nClade',
                       offset = 0.008, offset.text = 0.001,
                       fontsize = 3) +
       ggplot2::xlim(0, 0.09) +
       theme(text = element_text(family = 'Helvetica',
                                 colour = '#404040'))
p <- p %>% rotate(186) # to show nodes to rotate: + geom_text(aes(label = node))


# clade tree
c <- ggtree(tree, ladderize = FALSE, aes(color = group)) +
  geom_treescale(y = -6, offset = -4) +
  geom_tiplab(color = '#404040') +
  geom_nodelab(color = '#404040', hjust = -0.1) +
  scale_colour_manual(values = palette) +
  ggplot2::xlim(0, 0.075) +
  theme(text = element_text(family = 'Helvetica',
                            size = 3))
v <- viewClade(c, node = malagasy_clade) # view the malagasy clade
v <- v %>% rotate(362)

# overall plot
g <- plot_grid(p, v, # plot both overall tree and clade
               ncol = 2,
               labels = 'AUTO', # labels A and B
               label_fontfamily = 'Helvetica',
               label_colour = '#404040',
               label_fontface = 'plain', # no bold
               rel_widths = c(1.2, 1.8)) # relative widths of plots to each other

ggsave(g, file = file.path('..', 'results', 'figures', 'fullTree.pdf'))
```



Hypothesis Map Mini-Phylogeny

To create a mini-phylogeny to show the expected pattern of divergence, we can use the `ape` package to create a tree using parenthetical notation.

```
tree <- read.tree(text = '((Dry, Dry), (Dry/Humid, (Humid, Humid)));')

ggTree <- ggtree(tree) +
  geom_tiplab(hjust = -0.1) +
  ggplot2::xlim(c(0, 3.5)) + # provide xlims to prevent labels from running off page
  theme(text = element_text(family = 'Helvetica',
                            colour = '#404040',
                            size = 12))
ggTree <- ggTree %>% rotate(6)

ggsave(file.path('..', 'results', 'figures', 'hypTree.pdf'),
       width = 2, height = 2, units = 'in')
```


Humid
Humid
Dry/Humid
Dry
Dry