pollen_clusteR

Reed & Emily

4/10/2021

We want a spreadsheet for easy interactive entering of data. Because we are on a shared lab computer without alot of common software, we can't exactly go around installing things like R, and some microsoft suites. I really do not like the functionality of google sheets, so we made our own app to collect data into. This will allow us to drag these data back into the R session really quickly. We will also have a live- interactive display of multiple clustering matrices soon.

These are the libraries we use for the project.

```
library(tidyverse)
library(cluster)
library(shiny)
library(rhandsontable)
library(rsconnect)
library(ggplot2)
```

I quickly set up an Rstudio.io account, using the direction here: https://shiny.rstudio.com/articles/shinyapps.html?_ga=2.191076643.86591529.1618194183-1026877914.1618194183

Import Data to score

```
slides1 <- read.csv("L:\\Pollen_Reference_Library\\existing_pollen_reference_slides.csv")[,c(1:2,4:5)] %
  janitor::clean_names() %>%
  rename(genus = i_genus) %>%
  filter(type == 'Image') %>%
  select(-type)

#slides2 <- read.csv()
#pollen_df <- cbind(slides1, slides2)
head(slides1)
```

```
##            genus     epithet       family
## 1      Heracleum sphondylium     Apiaceae
## 2     Ligusticum     porteri     Apiaceae
## 3      Osmorhiza depauperata     Apiaceae
## 4        Senecio    wootonii   Asteraceae
## 5 Symphyotrichum   foliaceum   Asteraceae
## 6        Lappula   squarrosa Boraginaceae
```

```
slides2 <- readxl::read_excel("L:\\Pollen_Reference_Library\\Label_box.xlsx")[,3:4] %>%
  separate(Taxon, into = c("genus", "epithet"), sep = " ")
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 93 rows [1, 2, 3, 4,
## 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, ...].
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 6 rows [30, 35,
## 64, 82, 108, 109].
```

Interactive data scoring table.

We Will tag some empty columns onto our dataframe so that we places to insert our data.

```
pollen_df <- slides1 %>%
  mutate(outline = "") %>%
  mutate(polarity = "") %>%
  mutate(dispersal_unit = "") %>%
  mutate(aperturate = "") %>%
  mutate(ornamentation = "") %>%
  mutate(saccate = "") %>%
  mutate(operculate = "") %>%
  mutate(annulate = "") %>%
  mutate(aperture_type = "") %>%
  mutate(aperture_number = "") %>%
  mutate(l_polar_lat = "") %>%
  mutate(l_equi_lat = "")

pollen_df <- read.csv("L:\\Pollen_Reference_Library\\pollen_clusteR\\Pollen_morpho_all.csv") %>%
  mutate(l_polar_lat = "") %>%
  mutate(l_equi_lat = "")
```

##Develop our glossary for drop-down options

```
outline_cat <- c("circular", "elliptic", "triangular", "lobate", "quadrangular", "polygonal")
polarity_cat <- c("heteropolar", "isopolar")
dispersal_unit_cat <- c("monad", "dyad", "tryad", "polyad")
aperturate_cat <- c("none", "heteroaperturate", "angulaperturate", "planaperturate")
ornamentation_cat <- c("areolate", "bacculate", "bireticulate", "clavate", "clypteate", "echinate", "fo
saccate_cat <- c("none", "monosaccate", "bisaccate", "trisaccate")
operculate_cat <- c("yes", "no")
annulate_cat <- c("yes", "no")
apertures_cat <- c("porate", "colpate", "colporate", "syncolpate")
aperture_num_cat <- c("mono", "di", "tri", "tetra", "penta", "stephano", "panto")
```

Analysis Section

Here we implement the app.

```
color_renderer <- "function(instance, td) {
    Handsontable.renderers.TextRenderer.apply(this, arguments);
    td.style.color = 'black';
  }" # defining that we want BLACK text in our table.

ui <- shinyUI(fluidPage(
  fluidRow(
    titlePanel(title = "Pollen Morphology Scoring Sheet"),
    hr(),
    column(4,
           rHandsontableOutput("table"),
           br(),
           actionButton("saveBtn", "Save")
```

```
  ))
)) # creating the user interface, KISS.

server <- shinyServer(function(input, output, session){

  output$table <- renderRHandsontable({
    rhandsontable(pollen_df, width = 1400, height = 750) %>% #make the table big!!!
      hot_col(1:15, renderer = color_renderer,) %>% # make the damn text black.
      hot_col(1:3, readOnly = TRUE) %>%  # disable ediing these values
      hot_table(highlightCol = TRUE, highlightRow = TRUE) %>% #let us see where we are on the spreadhss

      hot_col(col = "outline", type = "dropdown", source = outline_cat, strict = F) %>% # dropdown opti
      hot_col(col = "polarity", type = "dropdown", source = polarity_cat, strict = F) %>%
      hot_col(col = "dispersal_unit", type = "dropdown", source = dispersal_unit_cat, strict = F) %>%
      hot_col(col = "aperturate", type = "dropdown", source = aperturate_cat, strict = F) %>%
      hot_col(col = "ornamentation", type = "dropdown", source = ornamentation_cat, strict = F) %>%
      hot_col(col = "saccate", type = "dropdown", source = saccate_cat, strict = F) %>%
      hot_col(col = "operculate", type = "dropdown", source = operculate_cat, strict = F) %>%
      hot_col(col = "annulate", type = "dropdown", source = annulate_cat, strict = F) %>%
      hot_col(col = "aperture_type", type = "dropdown", source = apertures_cat, strict = F) %>%
      hot_col(col = "aperture_number", type = "dropdown", source = aperture_num_cat, strict = F) %>%

      hot_col(col = "l_polar_lat", type = "numeric",  strict = F) %>%
      hot_col(col = "l_equi_lat", type = "numeric", strict = F)
  })

  observeEvent(input$saveBtn, write.csv(hot_to_r(input$table), file = paste0("Pollen_morpho_matrix.", Sy

}) # And here is our table, we have


shinyApp(ui = ui, server = server)


rm(annulate_cat, aperturate_cat, aperture_num_cat, apertures_cat, color_renderer, dispersal_unit_cat, op
```

You may go back and iteratively add more rows to the DF, or deal with some troublesome grains this way.

```
raw <- list.files(pattern = "^Pollen_morpho_matrix")
pollen_df <- read.csv(raw)
```

Clustering

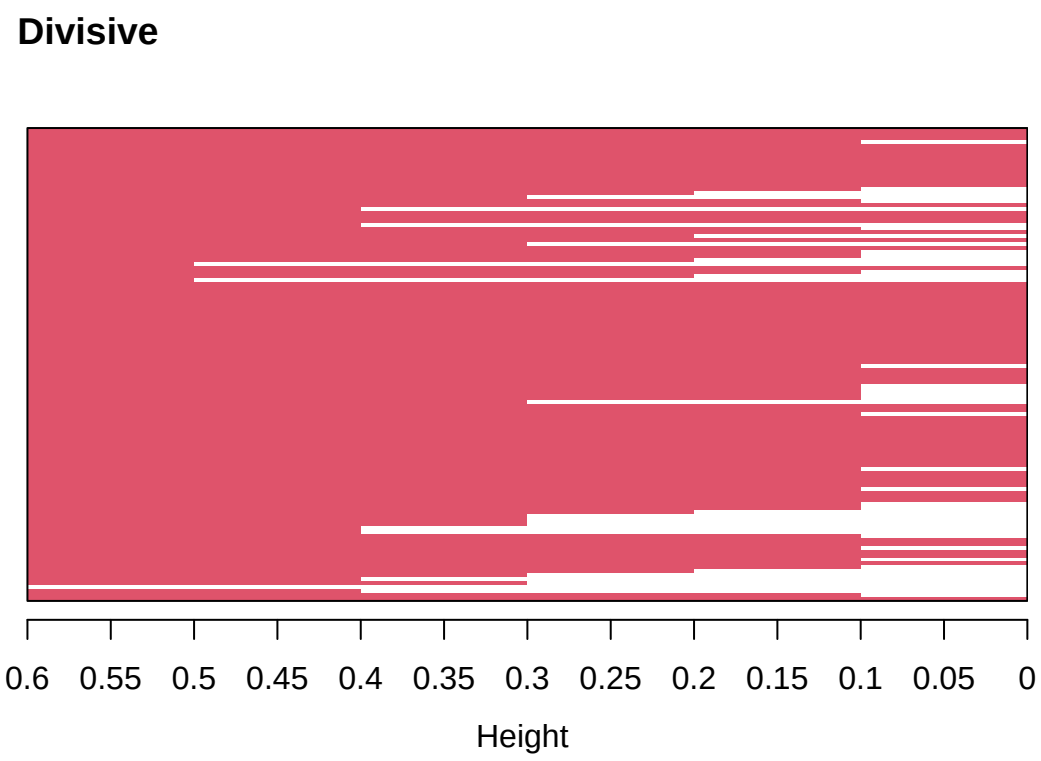
Matrices & Calculations

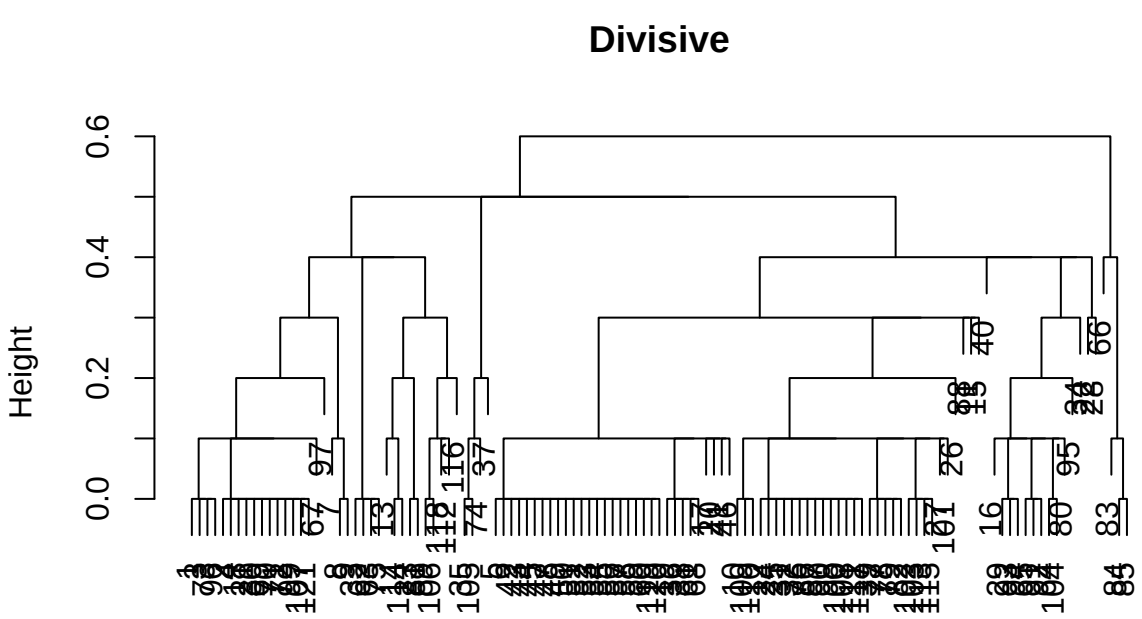
```
raw <- list.files(pattern = "^Pollen_morpho_matrix")
scored_data <- read.csv(raw[5], na.strings=c("","NA")) %>% janitor::clean_names()
rm(raw)
```

```
scored_data <- scored_data %>%
  filter(if_any(4:13, ~!is.na(.))) %>%
  mutate(across(4:13, ~  as.factor(.))) %>%
  select(-l_polar_lat, -l_equi_lat) # test purposes remove later.
```

```
gower.dist <- daisy(scored_data[ ,4:13], metric = c("gower"))
```

```
divisive.clust <- diana(as.matrix(gower.dist),
                  diss = TRUE, keep.diss = TRUE)

plot(divisive.clust, main = "Divisive")[1]
```






```
## NULL
```

Diagnostics

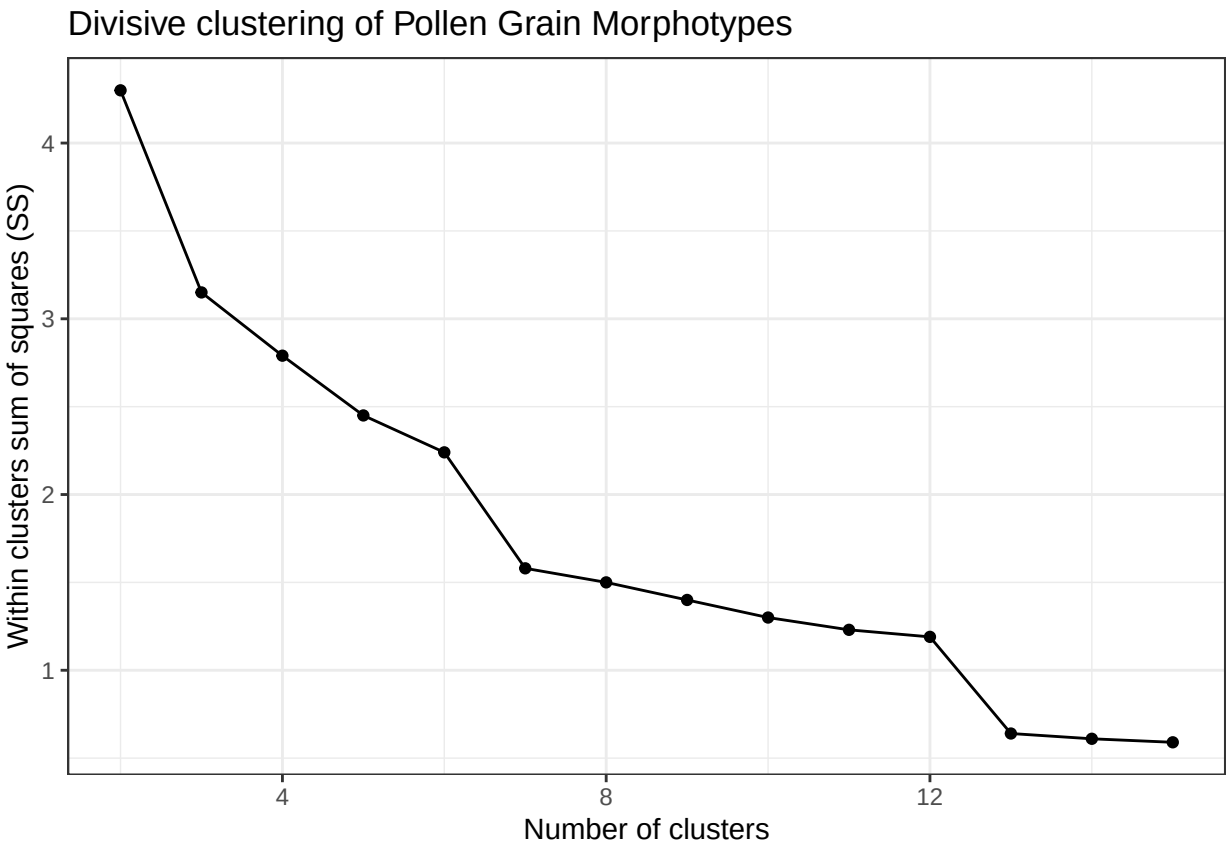
```
library(fpc)

cstats.table <- function(dist, tree, k) {
  clust.assess <- c("cluster.number","n","within.cluster.ss","average.within","average.between",
                "wb.ratio","dunn2","avg.silwidth")
  clust.size <- c("cluster.size")
  stats.names <- c()
  row.clust <- c()
  output.stats <- matrix(ncol = k, nrow = length(clust.assess))
  cluster.sizes <- matrix(ncol = k, nrow = k)

    for(i in c(1:k)){
      row.clust[i] <- paste("Cluster-", i, " size")
  }
    for(i in c(2:k)){
      stats.names[i] <- paste("Test", i-1)

    for(j in seq_along(clust.assess)){
      output.stats[j, i] <- unlist(cluster.stats(d = dist, clustering = cutree(tree, k =  i))[clust.ass

  }
```

```
  for(d in 1:k) {
    cluster.sizes[d, i] <- unlist(cluster.stats(d = dist, clustering = cutree(tree, k = i))[clust.size]
    dim(cluster.sizes[d, i]) <- c(length(cluster.sizes[i]), 1)
    cluster.sizes[d, i]

  }
}
  output.stats.df <- data.frame(output.stats)
  cluster.sizes <- data.frame(cluster.sizes)
  cluster.sizes[is.na(cluster.sizes)] <- 0
  rows.all <- c(clust.assess, row.clust)
  # rownames(output.stats.df) <- clust.assess
  output <- rbind(output.stats.df, cluster.sizes)[ ,-1]
  colnames(output) <- stats.names[2:k]
  rownames(output) <- rows.all
  is.num <- sapply(output, is.numeric)
  output[is.num] <- lapply(output[is.num], round, 2)
  output
}

stats.df.divisive <- cstats.table(gower.dist, divisive.clust, 7)
print(stats.df.divisive)
```

```
##                   Test 1 Test 2 Test 3 Test 4 Test 5 Test 6
## cluster.number      2.00   3.00   4.00   5.00   6.00   7.00
## n                 121.00 121.00 121.00 121.00 121.00 121.00
## within.cluster.ss   4.30   3.15   2.79   2.45   2.24   1.58
## average.within      0.24   0.20   0.19   0.17   0.16   0.14
## average.between     0.47   0.32   0.32   0.32   0.32   0.31
## wb.ratio            0.51   0.62   0.59   0.55   0.51   0.43
## dunn2               1.94   1.37   1.37   1.15   1.06   1.06
## avg.silwidth        0.49   0.34   0.36   0.32   0.34   0.43
## Cluster- 1  size  117.00  35.00  35.00  21.00  21.00  21.00
## Cluster- 2  size    4.00  82.00  78.00  78.00  78.00  63.00
## Cluster- 3  size    0.00   4.00   4.00  14.00  10.00  10.00
## Cluster- 4  size    0.00   0.00   4.00   4.00   4.00  15.00
## Cluster- 5  size    0.00   0.00   0.00   4.00   4.00   4.00
## Cluster- 6  size    0.00   0.00   0.00   0.00   4.00   4.00
## Cluster- 7  size    0.00   0.00   0.00   0.00   0.00   4.00
```

An elbow plot, of divisive clustering.

```
ggplot(data = data.frame(t(cstats.table(gower.dist, divisive.clust, 15))),
  aes(x=cluster.number, y=within.cluster.ss)) +
  geom_point()+
  geom_line()+
  ggtitle("Divisive clustering of Pollen Grain Morphotypes") +
  labs(x = "Number of clusters", y = "Within clusters sum of squares (SS)") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme_bw()
```



Visualization

```
library(dendextend)
```

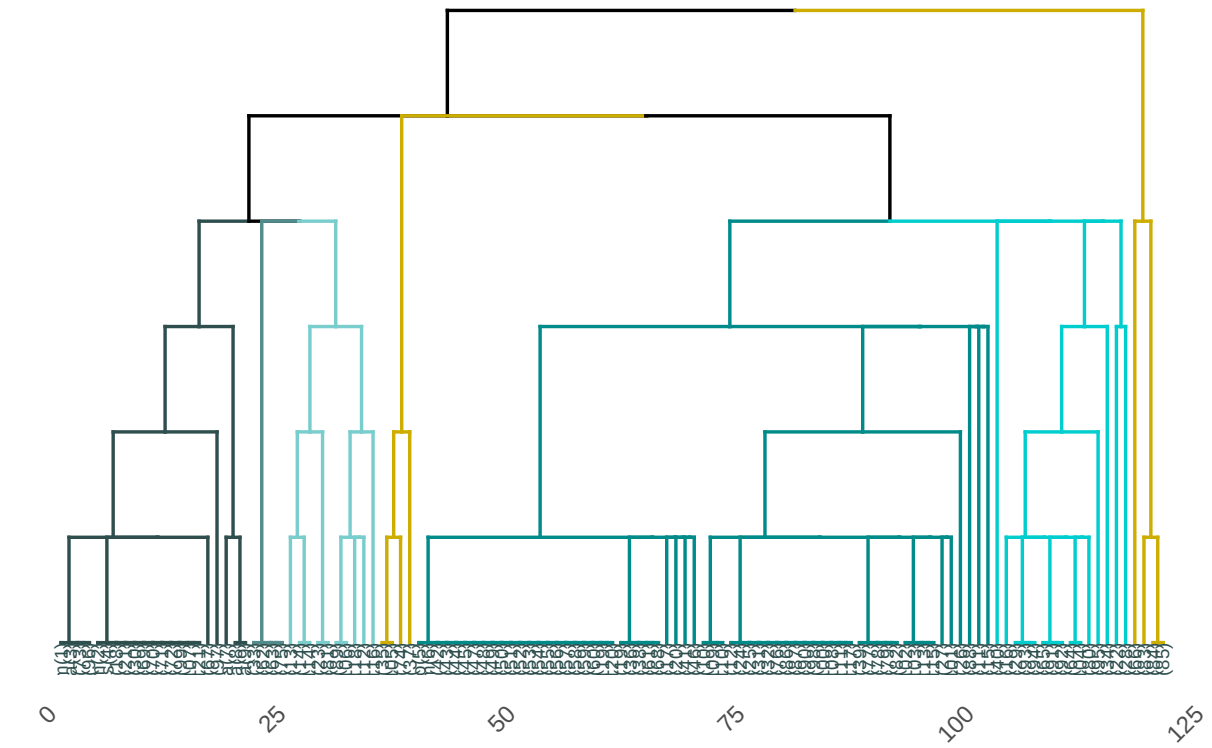
```
##
## ---------------------
## Welcome to dendextend version 1.14.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
dendro <- as.dendrogram(divisive.clust)
labels(dendro) <- paste(as.character(scored_data[,1])[order.dendrogram(dendro)], "(",labels(dendro),")"
```

```
# position <- scored_data %>%
#   rowid_to_column() %>%
#   pull(rowid)
#
# nomen <- scored_data %>%
#   unite(genus, col = 'binomial', epithet, sep = "_") %>%
#   pull(binomial)

# full_lookup <- setNames(nomen, position)
#divisive.clust$order <- full_lookup[divisive.clust[["order"]]]
#divisive.clust$order <- unname(divisive.clust[["order"]])

dendro.col <- dendro %>%
  set("branches_k_color", k = 7, value =  c("darkslategray", "darkslategray4", "darkslategray3", "gold3
  set("branches_lwd", 0.6) %>%
  set("labels_colors",
      value = c("darkslategray")) %>%
  set("labels_cex", 0.5)

ggd1 <- as.ggdend(dendro.col)

ggplot(ggd1) +
   labs(x = "Taxa", y = "Height", title = "Categorical Trait Based Dendrogram based on Gower Distances,
   theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 0.5))
```



```r
scored_data_1 <- scored_data %>%
  unite("binomial", genus:epithet)


png('cluster_analysis.png',  width = 3000, height = 3000, units = "px",)

par(cex = 2.5, mar = c(3,3,3,7))
plot(dendro,
     main = "Categorical Trait Based Dendrogram based on Gower Distances, k = 7",
     horiz =  TRUE,  nodePar = list(cex = 1.5))

dev.off()
```

```
## pdf
##   2
```