H1-T1: zona H1 (hybrid) del transecto 1.

José Carlos Del Valle

11/11/2021

Índice

1. Introducción.

En este script voy a analizar cómo ven las flores de *Erysimum merxmuelleri* y *E. lagascae* los principales grupos de polinizadores de la zona: dípteros, coleópteros, lepidópteros e himenópteros. Este script será un poco más largo porque es la primera zona con la que trabajo y he hecho varios test que no tengo que repetir en los siguientes scripts. Cada zona la anlizaré en scripts separados porque si no va a ser demasiado largo y un poco caótico

2. Importando y limpiando el dataset.

Los datos de reflectancia de la zona T1-H1 están en la carpeta "T1_Hibrida_101-178". En esa carpeta encontraremos dos mediciones: una hecha para la parte distal y otra para la parte apical. Nosotros utilizaremos el promedio de ambas mediciones, obteniendo tan solo una medición por flor.

Lo primero que haremos será cargar la librería *pavo*. Luego, cargaremos los datos (seleccionar el directorio apropiado) y crearemos el objeto **plotH1**.

```
library(pavo)
H1 <- "/Users/delValle/Dropbox/UGR/T1_Hibrida_101-178"
plotH1<-getspec(H1)
```

Ahora hay que pulir un poco los espectros en bruto que se obtienen del espectrofotómetro. Para ello utilizaremos las funciones "procspec" contenidas en el paquete *pavo*.

Lo primero que voy a hacer es eliminar posibles valores negativos de los espectros añadiendo el valor absoluto del valor negativo más alto a todo el espectro. A continuación, suavizo un poco los espectros.

```
plotH1.fix <- procspec(plotH1, fixneg = "addmin")
```

```
## processing options applied:
## Negative value correction: added min to all reflectance
```

```
plotH1.sm <- procspec(plotH1.fix, opt = "smooth", span = 0.2)
```

```
## processing options applied:
## smoothing spectra with a span of 0.2
```

```
is.rspec(plotH1.sm)
```

```
## [1] TRUE
```

Ahora voy a plotear los dos espectros del individuo 101 (101a = parte apical del pétalo, 101b = parte basal del pétalo).

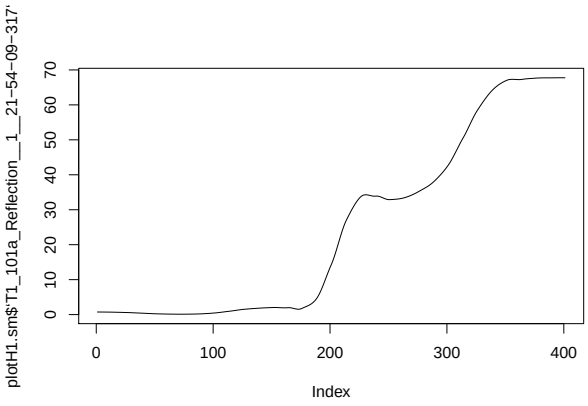

Figura 1: Parte apical de la flor del individuo 101.

Después de ver los espectros, no tengo demasiado claro si es buena idea mezclar las mediciones. Difieren demasiado entre ellos. De todas formas, voy a continuar así y calcularé el promedio de ambos espectros por individuo.

3. Obteniendo los promedios de las dos mediciones hechas en las partes basal y apical de la flor.

3.1. Primer test con un dataset reducido.

Para promediar los espectros de las partes basal y apical de una flor, hay que calcular el promedio de dos columnas contiguas en *plotH1.sm* de cada muestra. Voy a hacer primero un pequeño test de forma manual utilizando varias columnas continuas, lo que me servirá como referencia más adelante para verificar que lo que estoy haciendo está bien

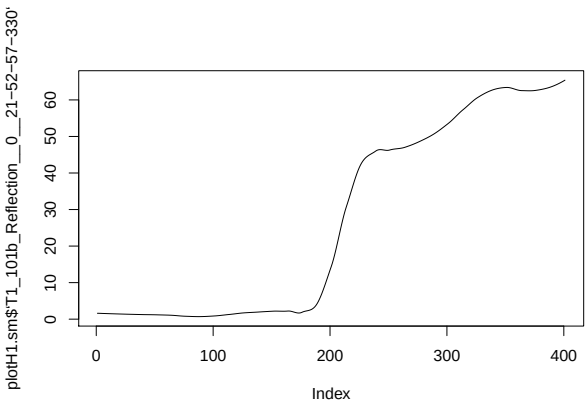

Figura 2: Parte basal de la flor del individuo 101.

```
plotH1.sm_manual <- plotH1.sm$wl
plotH1.sm_manual <- as.data.frame(plotH1.sm_manual)
plotH1.sm_manual$T1_101 <- apply(plotH1.sm[ ,c(2,3)], 1, mean, na.rm = TRUE)
plotH1.sm_manual$T1_102 <- apply(plotH1.sm[ ,c(4,5)], 1, mean, na.rm = TRUE)
plotH1.sm_manual$T1_103 <- apply(plotH1.sm[ ,c(6,7)], 1, mean, na.rm = TRUE)
plotH1.sm_manual$T1_104 <- apply(plotH1.sm[ ,c(8,9)], 1, mean, na.rm = TRUE)
```

Lo primero que he hecho ha sido crear una primera columna con las longitudes de onda. He transformado el objeto *plotH1.sm_manual* a data frame y a continuación he ido añadiendo columnas (una por individuo) con los promedios de ambas mediciones. Con una simple comprobación, vemos que el objeto *plotH1.sm_manual* es reconocido correctamente como un objeto rspec (formato de *pavo*):

```
plotH1.sm_manual <- as.rspec(plotH1.sm_manual)
```

```
## wavelengths found in column 1
```

```
is.rspec(plotH1.sm_manual)
```

```
## [1] TRUE
```

Estos serían los cinco primeros valores que he obtenido para las muestras 101 a 104:

```
head(plotH1.sm_manual, n=5)
```

```
##    wl   T1_101   T1_102   T1_103    T1_104
## 1 300 1.177922 1.302502 1.022938 0.6922963
## 2 301 1.171390 1.296603 1.026394 0.7040755
## 3 302 1.164755 1.290602 1.029493 0.7153003
## 4 303 1.158008 1.284509 1.032242 0.7259692
## 5 304 1.151145 1.278331 1.034647 0.7360804
```

Como es inviable hacerlo manualmente, voy a crear un loop para que automáticamente me calcule el promedio cada dos columnas. Primero voy a correr un primer test con los promedios desde la columna 2 a la 11:

```
test0 <- plotH1.sm$wl
test0 <- as.data.frame(test0)
colnames(test0) <- c("wl")
head(test0)
```

```
##    wl
## 1 300
## 2 301
## 3 302
## 4 303
## 5 304
## 6 305
```

```
for(i in 2:11) {
  test0[ , i] <- apply(plotH1.sm[ , c(i, i+1)], 1, mean, na.rm=TRUE)
}
```

Al comparar los resultados con los obtenidos de forma manual (ver objeto *plotH1.sm_manual*), veo que me ha sumado las columnas contiguas (ej: columna 2 + columna 3, columna 3 + columna 4, columna 4 + columna 5):

```
test0[1:5, 1:8]
```

```
##    wl       V2       V3       V4       V5       V6        V7        V8
## 1 300 1.177922 1.606348 1.302502 1.052695 1.022938 0.8655964 0.6922963
## 2 301 1.171390 1.594218 1.296603 1.053359 1.026394 0.8749099 0.7040755
## 3 302 1.164755 1.582198 1.290602 1.053727 1.029493 0.8837075 0.7153003
## 4 303 1.158008 1.570291 1.284509 1.053809 1.032242 0.8920017 0.7259692
## 5 304 1.151145 1.558501 1.278331 1.053614 1.034647 0.8998051 0.7360804
```

```
plotH1.sm_manual[1:5, 1:5]
```

```
##    wl   T1_101   T1_102   T1_103    T1_104
## 1 300 1.177922 1.302502 1.022938 0.6922963
## 2 301 1.171390 1.296603 1.026394 0.7040755
## 3 302 1.164755 1.290602 1.029493 0.7153003
## 4 303 1.158008 1.284509 1.032242 0.7259692
## 5 304 1.151145 1.278331 1.034647 0.7360804
```

Al sumar las columnas contiguas, únicamente las columnas pares (V2, V4, V6, etc.) son las correctas. Es decir, V2 equivale a T1_101, V4 equivale a T1_102, V6 equivale a T1_103 y V8 equivale a T1_104.

Voy a probar a eliminar las columnas impares y a retener las pares, pero antes tengo que quitar la primera columna, que es la que tiene las longitudes de onda.

```
test1 <- test0[ , -c(1)]
col_odd <- seq_len(ncol(test1)) %% 2
test1 <- test1[ , col_odd == 1]
rm(col_odd)
```

Ahora que solo tengo las columnas pares (las que tienen los promedios que estoy buscando), vuelvo a añadir la columna con las longitudes de onda (columna *wl*) que había eliminado. Ahora sí que me coinciden los resultados:

```
test1$wl <- test0$wl
library(dplyr)
test1 <- test1 %>%
  select(wl, everything())
```

Ahora sí que me coinciden los resultados:

```
plotH1.sm_manual[1:5, 1:5]
```

```
##    wl   T1_101   T1_102   T1_103    T1_104
## 1 300 1.177922 1.302502 1.022938 0.6922963
## 2 301 1.171390 1.296603 1.026394 0.7040755
## 3 302 1.164755 1.290602 1.029493 0.7153003
## 4 303 1.158008 1.284509 1.032242 0.7259692
## 5 304 1.151145 1.278331 1.034647 0.7360804
```

```
test1[1:5, 1:5]
```

```
##    wl       V2       V4       V6        V8
## 1 300 1.177922 1.302502 1.022938 0.6922963
## 2 301 1.171390 1.296603 1.026394 0.7040755
## 3 302 1.164755 1.290602 1.029493 0.7153003
## 4 303 1.158008 1.284509 1.032242 0.7259692
## 5 304 1.151145 1.278331 1.034647 0.7360804
```

```
rm(test0)
```

Ahora solo me faltaría cambiar el nombre de las variables y ya estaría listo:

```
x0 <- c("wl")
x1 <- 101:125
x2 <- 127:150
x3 <- 152:158
x4 <- 160:178
```

Como faltan algunas muestras (ej: 126), las secuencias no van del 101 al 178 directamente, sino que he creado objetos x con una numeración acorde al nombre de las muestras. Solo tengo que unificar estos objetos y añadirle al nombre de cada individuo (i.e. el nombre de cada columna) el prefijo "T1_" correspondiente al transecto

```
x_sum <- c(x1, x2, x3, x4)
x_sum <- paste("T1", x_sum, sep="_")
head(x_sum)
```

```
## [1] "T1_101" "T1_102" "T1_103" "T1_104" "T1_105" "T1_106"
```

Todo esto que he hecho serviría para tener los nombres de todas las muestras, pero para este pequeño test solo necesito los nombres desde el T_101 hasta el T1_105:

```
x_sum_test1 <- c(x_sum[1:5])
names_test1 <- c(x0,x_sum_test1)
names_test1
```

```
## [1] "wl"     "T1_101" "T1_102" "T1_103" "T1_104" "T1_105"
```

Ya tengo listo el vector con los nombres de las columnas del dataframe *test1*

```
colnames(test1) <- names_test1
head(test1, n=5)
```

```
##    wl   T1_101   T1_102   T1_103    T1_104   T1_105
## 1 300 1.177922 1.302502 1.022938 0.6922963 1.303106
## 2 301 1.171390 1.296603 1.026394 0.7040755 1.315932
## 3 302 1.164755 1.290602 1.029493 0.7153003 1.328119
## 4 303 1.158008 1.284509 1.032242 0.7259692 1.339655
## 5 304 1.151145 1.278331 1.034647 0.7360804 1.350528
```

Así es como consigo obtener los valores promedios de cada muestra (T1_101, T1_102, etc.) con su correspondiente nomenclatura; con esto podría trabajar ya y extrapolarlo al conjunto completo de datos.

Elimino todo aquello que ya no necesito antes de crear un data frame con todos los datos del transecto 1 de la zona híbrida H1.

```
rm(x1)
rm(x2)
rm(x3)
rm(x4)
rm(x0)
rm(x_sum)
rm(x_sum_test1)
rm(test1)
rm(names_test1)
rm(i)
```

3.2. Opción 1: para hacerlo de la forma que lo he hecho anteriormente (si no comprendes algún paso, arriba se explica paso a paso para qué se hace cada cosa).

```
plotH1_average <- plotH1.sm$wl
plotH1_average <- as.data.frame(plotH1_average)
colnames(plotH1_average) <- c("wl")
head(plotH1_average)
```

```
##    wl
## 1 300
## 2 301
## 3 302
## 4 303
## 5 304
## 6 305
```

```r
for(i in 2:150) {
  plotH1_average[ , i] <- apply(plotH1.sm[ , c(i, i+1)], 1, mean, na.rm=TRUE)
}
plotH1_average[1:5,1:5]
```

```
##    wl       V2       V3       V4       V5
## 1 300 1.177922 1.606348 1.302502 1.052695
## 2 301 1.171390 1.594218 1.296603 1.053359
## 3 302 1.164755 1.582198 1.290602 1.053727
## 4 303 1.158008 1.570291 1.284509 1.053809
## 5 304 1.151145 1.558501 1.278331 1.053614
```

```r
plotH1_average[1:5,145:150]
```

```
##       V145     V146     V147     V148     V149      V150
## 1 1.567030 3.450058 3.926525 2.167462 1.556768 0.7105921
## 2 1.564825 3.435370 3.912730 2.169318 1.557202 0.7169350
## 3 1.562205 3.420320 3.898625 2.170780 1.557302 0.7227992
## 4 1.559153 3.404901 3.884223 2.171847 1.557066 0.7281846
## 5 1.555651 3.389105 3.869535 2.172519 1.556487 0.7330916
```

```r
plotH1_average2 <- plotH1_average[ , -c(1)]
col_odd <- seq_len(ncol(plotH1_average2)) %% 2
plotH1_average2 <- plotH1_average2[ , col_odd == 1]
rm(col_odd)

plotH1_average2$wl <- plotH1_average$wl
library(dplyr)
plotH1_average2 <- plotH1_average2 %>%
  select(wl, everything())
plotH1_average2[1:4,1:5]
```

```
##    wl       V2       V4       V6        V8
## 1 300 1.177922 1.302502 1.022938 0.6922963
## 2 301 1.171390 1.296603 1.026394 0.7040755
## 3 302 1.164755 1.290602 1.029493 0.7153003
## 4 303 1.158008 1.284509 1.032242 0.7259692
```

```r
x0 <- c("wl")
x1 <- 101:125
x2 <- 127:150
x3 <- 152:158
x4 <- 160:178
x_sum <- c(x1, x2, x3, x4)
x_sum <- paste("T1", x_sum, sep="_")
head(x_sum)
```

```
## [1] "T1_101" "T1_102" "T1_103" "T1_104" "T1_105" "T1_106"
```

```
names_H1 <- c(x0,x_sum)
names_H1
```

```
##  [1] "wl"     "T1_101" "T1_102" "T1_103" "T1_104" "T1_105" "T1_106" "T1_107"
##  [9] "T1_108" "T1_109" "T1_110" "T1_111" "T1_112" "T1_113" "T1_114" "T1_115"
## [17] "T1_116" "T1_117" "T1_118" "T1_119" "T1_120" "T1_121" "T1_122" "T1_123"
## [25] "T1_124" "T1_125" "T1_127" "T1_128" "T1_129" "T1_130" "T1_131" "T1_132"
## [33] "T1_133" "T1_134" "T1_135" "T1_136" "T1_137" "T1_138" "T1_139" "T1_140"
## [41] "T1_141" "T1_142" "T1_143" "T1_144" "T1_145" "T1_146" "T1_147" "T1_148"
## [49] "T1_149" "T1_150" "T1_152" "T1_153" "T1_154" "T1_155" "T1_156" "T1_157"
## [57] "T1_158" "T1_160" "T1_161" "T1_162" "T1_163" "T1_164" "T1_165" "T1_166"
## [65] "T1_167" "T1_168" "T1_169" "T1_170" "T1_171" "T1_172" "T1_173" "T1_174"
## [73] "T1_175" "T1_176" "T1_177" "T1_178"
```

```
colnames(plotH1_average2) <- names_H1
dim(plotH1_average2)
```

```
## [1] 401  76
```

Las dimensiones del objeto *plotH1_average2* son correctas. Tiene 401 filas (el encabezado + 400nm) y 76 columnas (wl + 75 plantas).

```
head(plotH1_average2, n=5)
```

```
##    wl   T1_101   T1_102   T1_103    T1_104   T1_105   T1_106    T1_107   T1_108
## 1 300 1.177922 1.302502 1.022938 0.6922963 1.303106 1.111165 0.6805735 1.049051
## 2 301 1.171390 1.296603 1.026394 0.7040755 1.315932 1.126826 0.6778195 1.059807
## 3 302 1.164755 1.290602 1.029493 0.7153003 1.328119 1.141892 0.6750776 1.069984
## 4 303 1.158008 1.284509 1.032242 0.7259692 1.339655 1.156353 0.6723518 1.079580
## 5 304 1.151145 1.278331 1.034647 0.7360804 1.350528 1.170199 0.6696463 1.088595
##      T1_109    T1_110    T1_111   T1_112   T1_113   T1_114   T1_115   T1_116
## 1 0.6951392 0.9393013 0.8598050 1.193591 1.815666 1.543634 3.211835 2.066130
## 2 0.7030502 0.9477575 0.8596325 1.198291 1.813789 1.538387 3.195573 2.065815
## 3 0.7105574 0.9557559 0.8592762 1.202577 1.811728 1.533140 3.179492 2.065081
## 4 0.7176594 0.9632901 0.8587369 1.206450 1.809499 1.527897 3.163605 2.063917
## 5 0.7243551 0.9703539 0.8580152 1.209911 1.807115 1.522662 3.147928 2.062312
##      T1_117   T1_118    T1_119    T1_120   T1_121   T1_122   T1_123   T1_124
## 1 0.6163769 1.170082 0.5742310 0.9751733 1.128549 1.135917 1.202241 1.265450
## 2 0.6237399 1.164253 0.5735592 0.9753116 1.134356 1.138705 1.213282 1.252913
## 3 0.6307063 1.158455 0.5727570 0.9752445 1.139811 1.141374 1.223758 1.240575
## 4 0.6372709 1.152685 0.5718290 0.9749603 1.144916 1.143922 1.233674 1.228439
## 5 0.6434286 1.146945 0.5707795 0.9744474 1.149673 1.146351 1.243037 1.216506
##      T1_125   T1_127    T1_128    T1_129   T1_130   T1_131    T1_132   T1_133
## 1 0.7419735 1.971802 0.9656343 0.8741431 1.281989 1.415564 0.9023117 1.398973
## 2 0.7528430 1.971278 0.9606595 0.8840403 1.291282 1.407531 0.9089544 1.399520
## 3 0.7632717 1.970364 0.9555544 0.8934233 1.300068 1.399517 0.9151778 1.399876
## 4 0.7732540 1.969051 0.9503283 0.9022970 1.308355 1.391522 0.9209711 1.400039
## 5 0.7827843 1.967334 0.9449905 0.9106664 1.316148 1.383549 0.9263230 1.400008
##     T1_134    T1_135    T1_136    T1_137   T1_138   T1_139   T1_140    T1_141
## 1 2.116579 0.6915084 0.5532602 0.6905636 2.604496 2.531151 1.009144 0.8764612
## 2 2.108159 0.6988810 0.5670744 0.7017840 2.598581 2.519168 1.008649 0.8815723
```

```
## 3 2.099781 0.7058610 0.5803072 0.7125795 2.592402 2.507134 1.008017 0.8863190
## 4 2.091449 0.7124579 0.5929559 0.7229494 2.585962 2.495057 1.007241 0.8906949
## 5 2.083170 0.7186807 0.6050180 0.7328928 2.579264 2.482942 1.006313 0.8946936
##      T1_142   T1_143   T1_144    T1_145   T1_146   T1_147   T1_148   T1_149
## 1 0.8960364 1.485039 1.691036 0.4846095 1.914893 1.195676 1.229139 1.229258
## 2 0.9040141 1.486715 1.690417 0.4957352 1.912647 1.199611 1.235324 1.240328
## 3 0.9115291 1.488039 1.689754 0.5064356 1.910202 1.203182 1.241168 1.250989
## 4 0.9185812 1.489007 1.689054 0.5167225 1.907566 1.206378 1.246672 1.261230
## 5 0.9251700 1.489617 1.688322 0.5266076 1.904747 1.209186 1.251839 1.271037
##     T1_150   T1_152   T1_153    T1_154    T1_155    T1_156   T1_157   T1_158
## 1 1.935734 2.336883 1.270379 0.7658079 0.5256959 0.6141223 1.220436 1.498040
## 2 1.923138 2.326032 1.264073 0.7603560 0.5357716 0.6219688 1.219630 1.485949
## 3 1.910829 2.315192 1.257733 0.7548916 0.5453662 0.6295018 1.218729 1.474162
## 4 1.898810 2.304374 1.251352 0.7494088 0.5544733 0.6367201 1.217736 1.462684
## 5 1.887081 2.293584 1.244923 0.7439017 0.5630864 0.6436221 1.216653 1.451519
##     T1_160   T1_161    T1_162   T1_163    T1_164   T1_165   T1_166    T1_167
## 1 2.030380 1.406037 0.5245388 1.538550 0.2780937 2.623936 2.380977 0.9350185
## 2 2.013655 1.399601 0.5316905 1.558147 0.2860468 2.611667 2.374888 0.9442331
## 3 1.997084 1.393240 0.5384727 1.577063 0.2937263 2.599264 2.368606 0.9529143
## 4 1.980657 1.386953 0.5448726 1.595289 0.3011160 2.586716 2.362130 0.9610562
## 5 1.964363 1.380739 0.5508773 1.612816 0.3081992 2.574014 2.355459 0.9686534
##     T1_168    T1_169    T1_170    T1_171   T1_172   T1_173   T1_174   T1_175
## 1 1.463836 0.7604865 0.6090474 0.5345297 1.402610 1.079505 1.332258 1.289677
## 2 1.463481 0.7590971 0.6098860 0.5484804 1.411189 1.073245 1.342016 1.283073
## 3 1.462813 0.7576207 0.6105071 0.5617958 1.419202 1.066917 1.351244 1.276452
## 4 1.461823 0.7560570 0.6109124 0.5744683 1.426644 1.060533 1.359920 1.269821
## 5 1.460500 0.7544058 0.6111034 0.5864904 1.433509 1.054107 1.368023 1.263185
##     T1_176   T1_177    T1_178
## 1 3.450058 2.167462 0.7105921
## 2 3.435370 2.169318 0.7169350
## 3 3.420320 2.170780 0.7227992
## 4 3.404901 2.171847 0.7281846
## 5 3.389105 2.172519 0.7330916
```

```
plotH1_average2 <- as.rspec(plotH1_average2)
is.rspec(plotH1_average2)
```

```
## [1] TRUE
```

Eliminamos todo aquello que ya no necesito. Importante: *names_H1* me hará falta mas adelante, por eso no lo elimino todavía.

```
rm(x0)
rm(x1)
rm(x2)
rm(x3)
rm(x4)
rm(x_sum)
rm(i)
rm(plotH1_average)
```

3.2. Opción 2: para hacerlo más fino, tendría que crear un loop como el que describo abajo, pero no me termina de correr bien No lo incluyo en los chunks porque no me interesa correrlo ahora. Lo dejo como texto normal por si en un futuro soy capaz de solucionarlo.

plotH1_average <- plotH1.sm$wl
plotH1_average <- as.data.frame(plotH1_average)
colnames(plotH1_average) <- c("wl")
head(plotH1_average)

i_seq <- seq(from=2, to=151, by=2)

for(i in i_seq) {
plotH1_average[, i] <- apply(plotH1.sm[, c(i, i+1)], 1, mean, na.rm=TRUE)
}

plotH1_average
plotH1_average[1:5,1:5]
plotH1.sm_manual[1:5,1:5]

4. Cálculo de variables colorimétricas.

Las variables colorimétricas que pueden incluirse en el paper, tales como hue, chroma, spectral purity, etc., se calculan con este sencillo comando:

```
colouremetric.variables.plotH1_average2 <- summary(plotH1_average2)
# write.csv2(colouremetric.variables.plotH1_average2,
#            file="plotH1_average2_colourimetric_variables.csv", row.names = TRUE)
```

Se podría guardar en un csv los resultados, pero por el momento dejo las almohadillas porque no me interesa crear el archivo. Ahora voy a plotear los espectros de las 75 plantas de la zona H1. Además, la función colorea cada línea según la forma del espectro, tal y cómo se vería al ojo humano. Por último, voy a crear un objeto *rgb* con los colores de cada espectro, que utilizaré más adelante para colorear los puntos en los distintos modelos de visión.

```
plot(plotH1_average2, type = "o", col = spec2rgb(plotH1_average2))
```



```
rgb <- spec2rgb(plotH1_average2)
```

5. Modelos de visión.

5.1. Modelo de visión de las ABEJAS.

Para prevenir errores, voy a corregir posibles valores "cero" en nuestro dataset con la función *procspec*.

```
plotH1_average2 <- procspec(plotH1_average2, fixneg = "addmin")
```

Para representar las 75 flores provenientes de la zona H1 del transecto 1 en el hexágono de color (himenópteros), primero tengo que crear el modelo visual con los siguientes parámetros. He utilizado el "green" estándar como background, la iluminación D65, el quantum catch = Ei, el modelo visual de *Apis mellifera* y la correción de von kries.

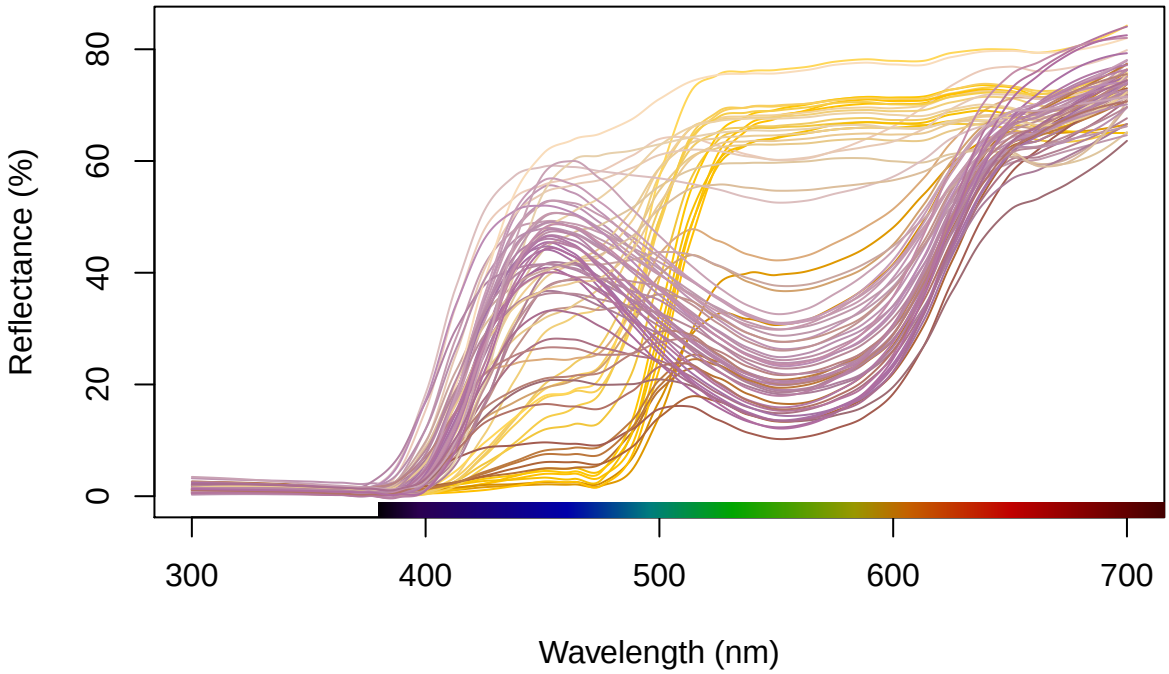

Figura 3: Espectros de todas las flores de la zona H1 del transecto 1.

```
vis.flowers.H1.bees <- vismodel(plotH1_average2,
                          visual = "apis",
                          qcatch = "Ei",
                          relative = FALSE,
                          vonkries = TRUE,
                          achromatic = "l",
                          bkg = "green",
                          illum="D65"
)
```

Una vez creado el modelo visual *vis.flowers.H1.bees*, ya puedo plotear los puntos en el hexágono.

```
hexagon.H1.bees <- colspace(vis.flowers.H1.bees, space = "hexagon")
hexplot(hexagon.H1.bees,
        sectors="coarse",
        labels=TRUE,
        main="Hybrid Transect 1",
        cex.main=1.5,
        col="black")
```

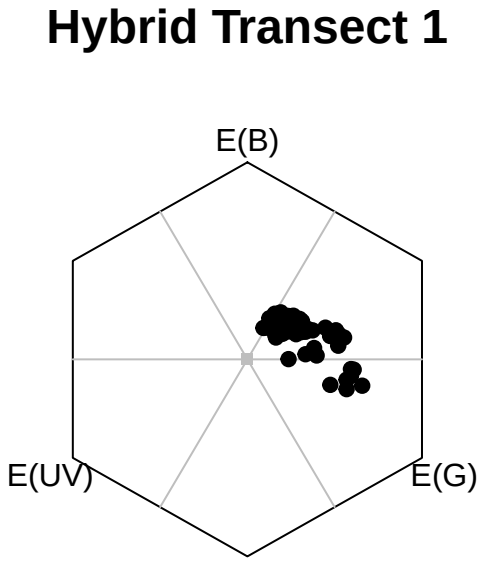

Figura 4: Hexágono de color con los datos en bruto.

Si quisiera guardar en un csv los valores obtenidos del hexágono, como por ejemplo la distancia al centro de cada punto, tendría que ejecutar el siguiente scrip:

```
# write.csv2(hexagon.H1.bees, file="hexagon.H1.bees.csv", row.names = TRUE)
```

Ahora voy a depurar el gráfico para poder exportarlo para la publicación, incluyendo los colores de cada flor tal y como se perciben al ojo humano. El gráfico lo exportaría en pdf a 7.5 x 5.5 pulgadas.

```
par(mar=c(1, 1, 1, 1))
hexplot(hexagon.H1.bees,
        sectors="coarse",
        achro=TRUE,
        labels=FALSE,
        cex=1.1,
        labels.cex = 1.2,
        col=rgb)
```

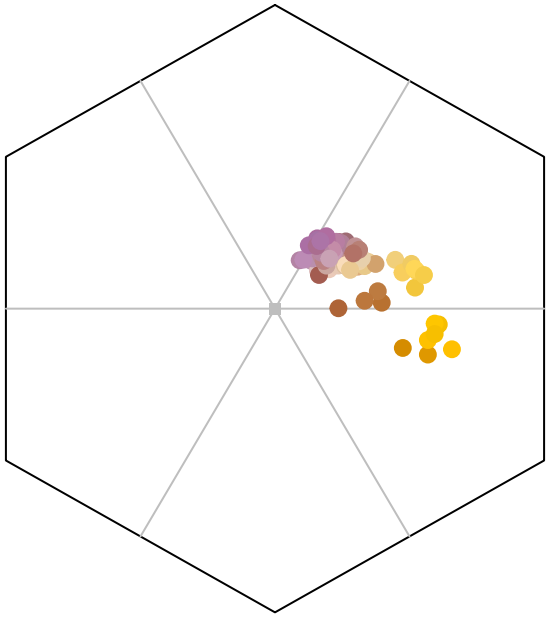

Figura 5: Hexágono de color con los color loci tal y como son percibidos por el ojo humano.