INF 550 Section 4.13

Natasha Wesely

2022-10-06

4.13 Digital Repeat Photography Coding Lab

4.13.1 Quantifying haze and redness to evaluate California wildfires

Question 1

Pull mid-day imagery for September 1-7th, 2019 and 2020 for the canopy-level camera NEON.D17.SOAP.DP1.00033. Create a 2-panel plot showing those images in 2019 (left) and 2020 (right).

```
# open a temporary directory
jpeg_dir <- "data_4_13"

# download a subset
download_midday_images(site = 'NEON.D17.SOAP.DP1.00033', # which site
                       y = 2019:2020, # which year(s)
                       months = 9, # which month(s)
                       days = 1:7, # which days on month(s)
                       download_dir = jpeg_dir) # where on your computer
```

```
##   |                                                                      |
```

```
## Warning in download_midday_images(site = "NEON.D17.SOAP.DP1.00033", y =
## 2019:2020, : data_4_13/NEON.D17.SOAP.DP1.00033_2019_09_02_120006.jpg was already
## in data_4_13
```

```
##   |                                                                      |==================
```

```
## Warning in download_midday_images(site = "NEON.D17.SOAP.DP1.00033", y =
## 2019:2020, : data_4_13/NEON.D17.SOAP.DP1.00033_2019_09_04_120006.jpg was already
## in data_4_13
```

```
##   |                                                                      |==================
```

```
## Warning in download_midday_images(site = "NEON.D17.SOAP.DP1.00033", y =
## 2019:2020, : data_4_13/NEON.D17.SOAP.DP1.00033_2019_09_06_120006.jpg was already
## in data_4_13
```

```
##   |                                                                      |==================
```

```
## Warning in download_midday_images(site = "NEON.D17.SOAP.DP1.00033", y =
## 2019:2020, : data_4_13/NEON.D17.SOAP.DP1.00033_2020_09_02_120006.jpg was already
## in data_4_13
```

```
##   |                                                                      |=================
```

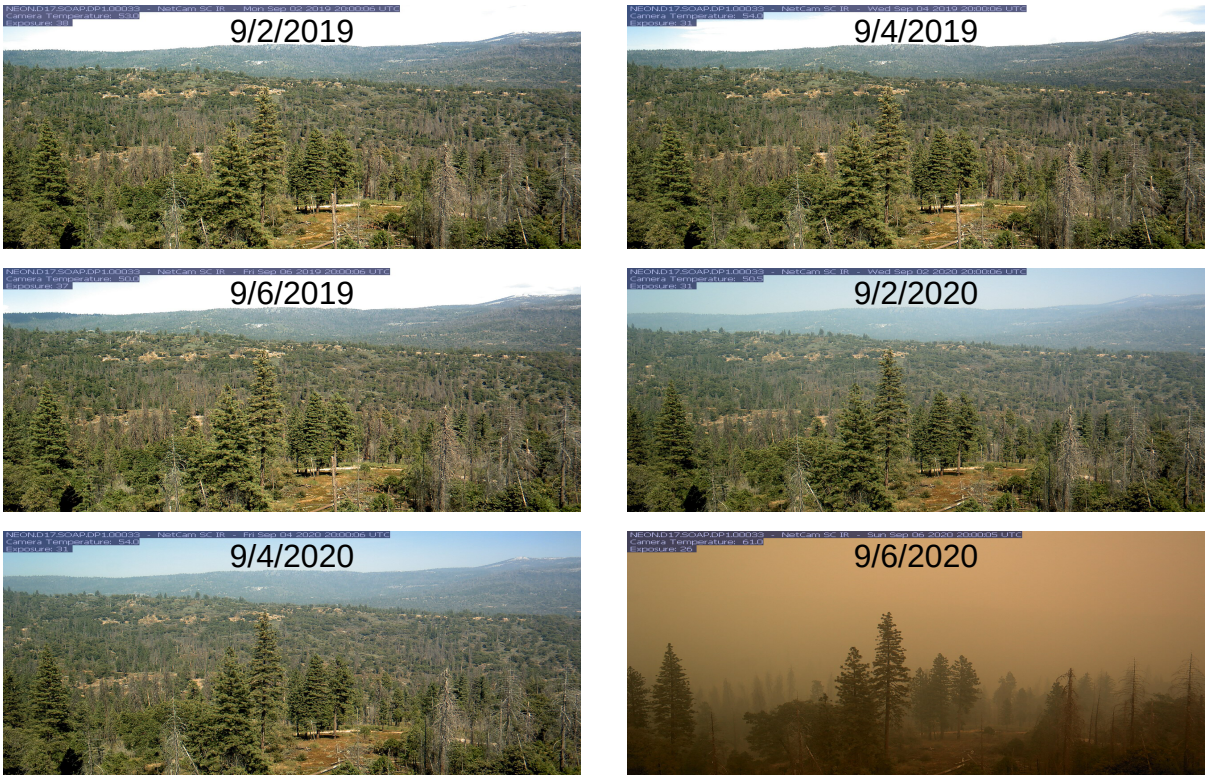
```
## Warning in download_midday_images(site = "NEON.D17.SOAP.DP1.00033", y =
## 2019:2020, : data_4_13/NEON.D17.SOAP.DP1.00033_2020_09_04_120006.jpg was already
## in data_4_13
```

```
##   |                                                                      |=================
```

```
## Warning in download_midday_images(site = "NEON.D17.SOAP.DP1.00033", y =
## 2019:2020, : data_4_13/NEON.D17.SOAP.DP1.00033_2020_09_06_120005.jpg was already
## in data_4_13
```

```
## [1] "data_4_13"
```

```r
# list of paths to the downloaded files
midday_paths <- dir(jpeg_dir, pattern = '.jpg*', full.names = TRUE)

# set up the plot dimensions
par(mar= c(0,0,0,0),
    mfrow=c(3,2),
    oma=c(0,0,3,0)
    )

# make a vector of the dates just to make the graphic nice
dates = c("9/2/2019","9/4/2019","9/6/2019","9/2/2020","9/4/2020","9/6/2020")

for (i in 1:length(midday_paths)) {
  img <- readJPEG(midday_paths[i])
  plot(0:1,0:1, type='n', axes= FALSE, xlab= '', ylab = '')
  rasterImage(img, 0, 0, 1, 1)
  mtext(dates[i], line = -2)
}
```



Question 2

Use the hazeR package to quantify the haze in each of those images. Print a summary of your results.

```
# create an empty matrix to fill with haze and A0 values
haze_mat <- data.frame()

# loop through each image and calc the haze factor
for (i in 1:length(midday_paths)){
  image_path <- midday_paths[i]
  img <- jpeg::readJPEG(image_path)
  # calc the haze level
  haze_degree <- getHazeFactor(img)
  # store the overall haze level for the image in the matix
  haze_mat <- rbind(haze_mat,
                    data.frame(file = as.character(image_path),
                               haze = haze_degree[1],
                               A0 = haze_degree[2]))

}

# Print a summary of your results
head(haze_mat)
```

```
##                                                    file      haze        A0
## 1 data_4_13/NEON.D17.SOAP.DP1.00033_2019_09_02_120006.jpg 0.3270243 0.6752243
## 2 data_4_13/NEON.D17.SOAP.DP1.00033_2019_09_04_120006.jpg 0.3107070 0.6553554
## 3 data_4_13/NEON.D17.SOAP.DP1.00033_2019_09_06_120006.jpg 0.3315426 0.6763943
## 4 data_4_13/NEON.D17.SOAP.DP1.00033_2020_09_02_120006.jpg 0.3146912 0.6634274
## 5 data_4_13/NEON.D17.SOAP.DP1.00033_2020_09_04_120006.jpg 0.2930479 0.6623760
## 6 data_4_13/NEON.D17.SOAP.DP1.00033_2020_09_06_120005.jpg 0.1378435 0.6651533
```

```
summary(haze_mat)
```

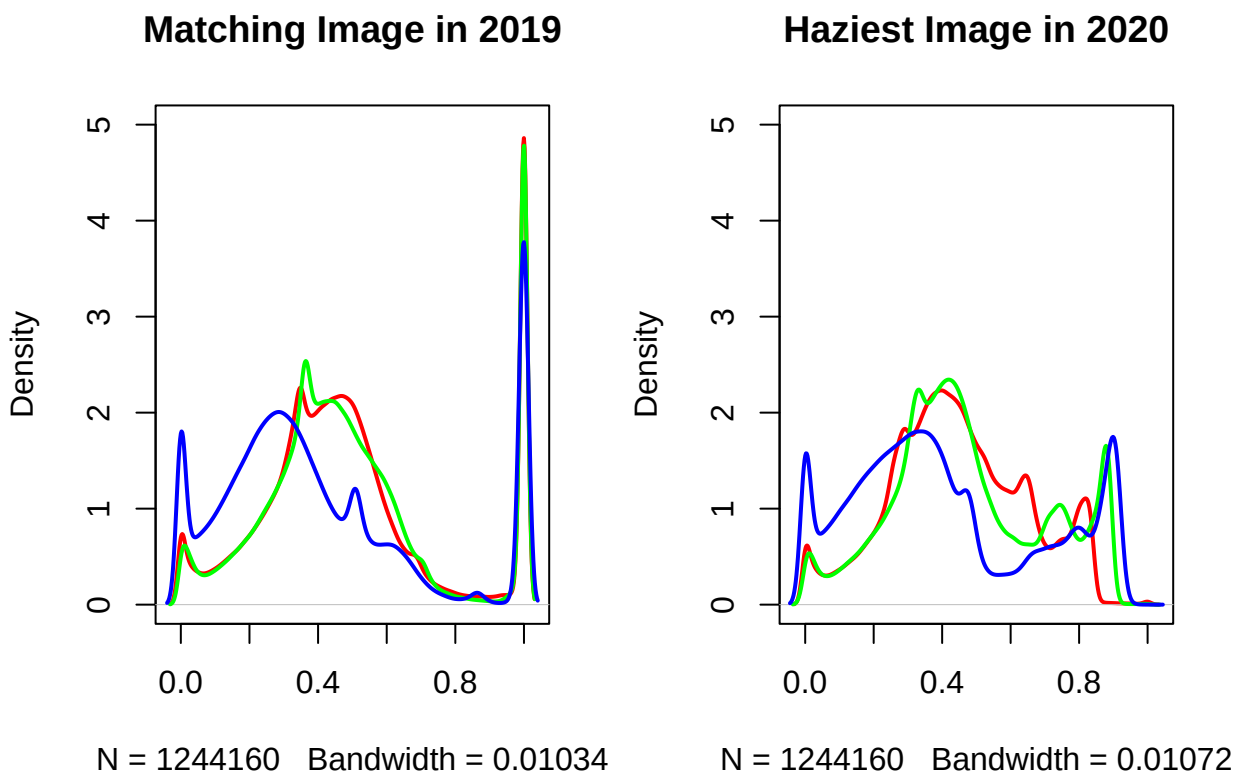
```
##      file                haze              A0
##  Length:6           Min.   :0.1378   Min.   :0.6554
##  Class :character   1st Qu.:0.2975   1st Qu.:0.6626
##  Mode  :character   Median :0.3127   Median :0.6643
##                     Mean   :0.2858   Mean   :0.6663
##                     3rd Qu.:0.3239   3rd Qu.:0.6727
##                     Max.   :0.3315   Max.   :0.6764
```

Question 3

Generate a density function RGB plot for your haziest image in 2020, and one for the same date in 2019. Create a 2-panel plot showing 2019 on the left and 2020 on the right.

```
# split the matrix by year
haze_mat2020 = haze_mat[which(grepl(pattern = "2020", x = haze_mat$file) == TRUE),]
haze_mat2019 = haze_mat[which(grepl(pattern = "2019", x = haze_mat$file) == TRUE),]

# which image is the most hazy in 2020? - Sept 2
mostHazy2020 = haze_mat2020[which(haze_mat2020$haze == max(haze_mat2020$haze)),]

# find the same day in 2019
match2019 = haze_mat2019[which(grepl(pattern = "09_02", x = haze_mat2019$file) == TRUE),]

# read the jpgs
rgb_array1 = jpeg::readJPEG(mostHazy2020[1,1])
rgb_array2 = jpeg::readJPEG(match2019[1,1])

# color channels extracted from the matrix
red_vector1 <- rgb_array1[,,1]
green_vector1 <- rgb_array1[,,2]
blue_vector1 <- rgb_array1[,,3]
red_vector2 <- rgb_array2[,,1]
green_vector2 <- rgb_array2[,,2]
blue_vector2 <- rgb_array2[,,3]

# plotting
par(mar=c(5,4,4,2), mfrow=c(1,2))
plot(density(red_vector2), col = 'red', lwd = 2,
     main = 'Matching Image in 2019', ylim = c(0,5))
lines(density(green_vector2), col = 'green', lwd = 2)
lines(density(blue_vector2), col = 'blue', lwd = 2)
plot(density(red_vector1), col = 'red', lwd = 2,
```

```
     main = 'Haziest Image in 2020', ylim = c(0,5))
lines(density(green_vector1), col = 'green', lwd = 2)
lines(density(blue_vector1), col = 'blue', lwd = 2)
```



Question 4

Pull timesseries data via the phenocamapi package. Calculate the difference in the rcc90 between 2019 and 2020 over the same time period as your images.

```
# define the date ranges
dateSeq1 = seq(ymd('2019-09-01'),ymd('2019-09-07'), by = '1 day')
dateSeq2 = seq(ymd('2020-09-01'),ymd('2020-09-07'), by = '1 day')
dateSeq = c(dateSeq1, dateSeq2)

# grab the data
data = read_csv("data_4_13/NEON.D17.SOAP.DP1.00033_EN_1000_1day.csv", skip = 24) %>%
  filter(date %in% dateSeq) %>%
  select(date, year, doy, rcc_90) %>%
  mutate(DayMonth = format(date, "%m-%d"))
```

```
## Rows: 1973 Columns: 49
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): midday_filename
```

```
## dbl  (46): year, doy, image_count, midday_r, midday_g, midday_b, midday_gcc,...
## lgl   (1): snow_flag
## date  (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
(RccDiffs = data %>%
  group_by(DayMonth) %>%
  summarise(Rcc_diff = diff(rcc_90)) %>%
    ungroup())
```

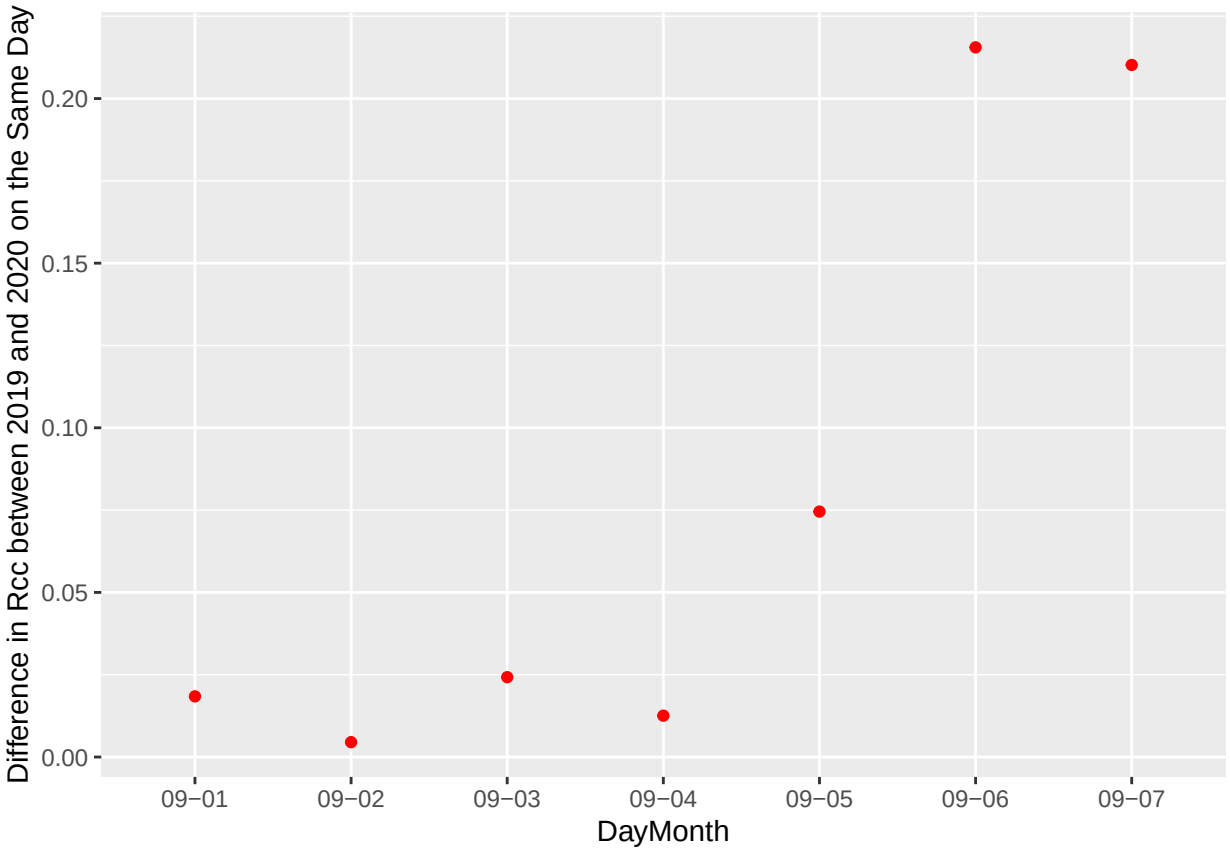
```
## # A tibble: 7 x 2
##   DayMonth Rcc_diff
##   <chr>       <dbl>
## 1 09-01     0.0184
## 2 09-02     0.00452
## 3 09-03     0.0243
## 4 09-04     0.0126
## 5 09-05     0.0746
## 6 09-06     0.216
## 7 09-07     0.210
```

The mean Rcc (90th percentile) in 2020 is 0.0800114 higher than the mean Rcc (90th percentile) in 2019.

Question 5

Create a summary plot showing haze as a bar and the difference in rcc90 from question 4 as a timerseries.

```
ggplot() +
  geom_point(data = RccDiffs, aes(x = DayMonth, y = Rcc_diff), color = "red") +
  labs(y = "Difference in Rcc between 2019 and 2020 on the Same Day")
```



Question 6

Answer the following questions:

Does the hazeR package pick up smokey images?

No, hazeR does not seem to pick up smokey images well.

If you were to use color coordinates, which color band would be most useful to highlight smoke and why?

The red band would be the most useful for detecting smoke in a photo because the smokey pixels have more red in them.