library(terra)

library(sf)

library(dplyr)

library(tidyr)

library(purrr)

library(exactextractr)

library(entropy)

library(beepr)

# ---- File paths ----

hsi\_folder <- "E:/HSI\_Files\_Parsing/"

shapefile\_folder <- "E:/Git Paint Rock 1.0/Updated Canopy Polygons/Updated/"

output\_csv <- "E:/DATA/HSI\_Metrics.csv"

# ---- Step 1: Load and filter filenames ----

shapefiles <- list.files(shapefile\_folder, pattern = "\\.shp$", full.names = TRUE)

shapefile\_numbers <- gsub("\\D", "", basename(shapefiles)) # Extract numbers from shapefile names

allfiles <- list.files(hsi\_folder)

hsi\_files\_all <- allfiles[!grepl("\\.hdr$|\\.aux$|\\.xml$|\\.enp$|\\.sta$", allfiles)] # Filter only .tif

img\_numbers <- gsub("\\D", "", hsi\_files\_all) # Extract numbers from HSI filenames

# ---- Step 2: Match shapefiles and HSI files by image number ----

matched\_files <- intersect(img\_numbers, shapefile\_numbers) # Only process where both exist

# Function to compute entropy

compute\_entropy <- function(x, bins = 10) {

counts <- hist(x, breaks = bins, plot = FALSE)$counts

if (sum(counts) == 0) return(NA)

entropy.empirical(counts, unit = "log2")

}

# ---- Step 4: Loop and extract ----

extracted\_list <- lapply(matched\_files, function(img\_number) {

cat("Processing image number:", img\_number, "\n")

# Match paths

img\_idx <- which(img\_numbers == img\_number)

shp\_idx <- which(shapefile\_numbers == img\_number)

if (length(img\_idx) == 0 || length(shp\_idx) == 0) return(NULL) # Skip if missing

img\_path <- file.path(hsi\_folder, hsi\_files\_all[img\_idx])

shp\_path <- shapefiles[shp\_idx]

# Load data

hsi\_raster <- rast(img\_path)

band\_names <- names(hsi\_raster)

canopy\_sf <- st\_read(shp\_path, quiet = TRUE)

# Extract values

extracted <- exact\_extract(

hsi\_raster,

canopy\_sf,

include\_cols = "Canopies",

progress = FALSE

)

polygon\_pixels\_df <- map\_df(extracted, function(df) {

df <- df %>% filter(!is.na(df[, 1]))

df$Canopies <- unique(df$Canopies)

df

})

polygon\_pixels\_df$ImageID <- img\_number

return(polygon\_pixels\_df)

})

beep()

# ---- Step 5: Combine and reshape ----

hsi\_pixels\_df <- bind\_rows(extracted\_list) %>%

separate(Canopies, into = c("TreeID", "SpeciesID"), sep = "\_", remove = TRUE)

# Long format

hsi\_long <- hsi\_pixels\_df %>%

pivot\_longer(

cols = matches("^\\d+\\.\\d+\\s\*nm$"), # Match spectral band names like "396.345 nm"

names\_to = "Wavelength",

values\_to = "Value"

) %>%

filter(!is.na(Value))

# Summarize per tree

hsi\_summary <- hsi\_long %>%

group\_by(TreeID, Wavelength) %>%

summarise(

Max = max(Value, na.rm = TRUE),

Min = min(Value, na.rm = TRUE),

Mean = mean(Value, na.rm = TRUE),

SD = sd(Value, na.rm = TRUE),

Entropy = compute\_entropy(Value),

.groups = "drop"

)

# Wide format

hsi\_wide <- hsi\_summary %>%

pivot\_wider(

names\_from = Wavelength,

values\_from = c(Max, Min, Mean, SD, Entropy),

names\_glue = "{.value}\_{Wavelength}"

)

# Export

write.csv(hsi\_wide, output\_csv, row.names = FALSE)

beep()