help_otsuSeg.R

achour

2025-03-29

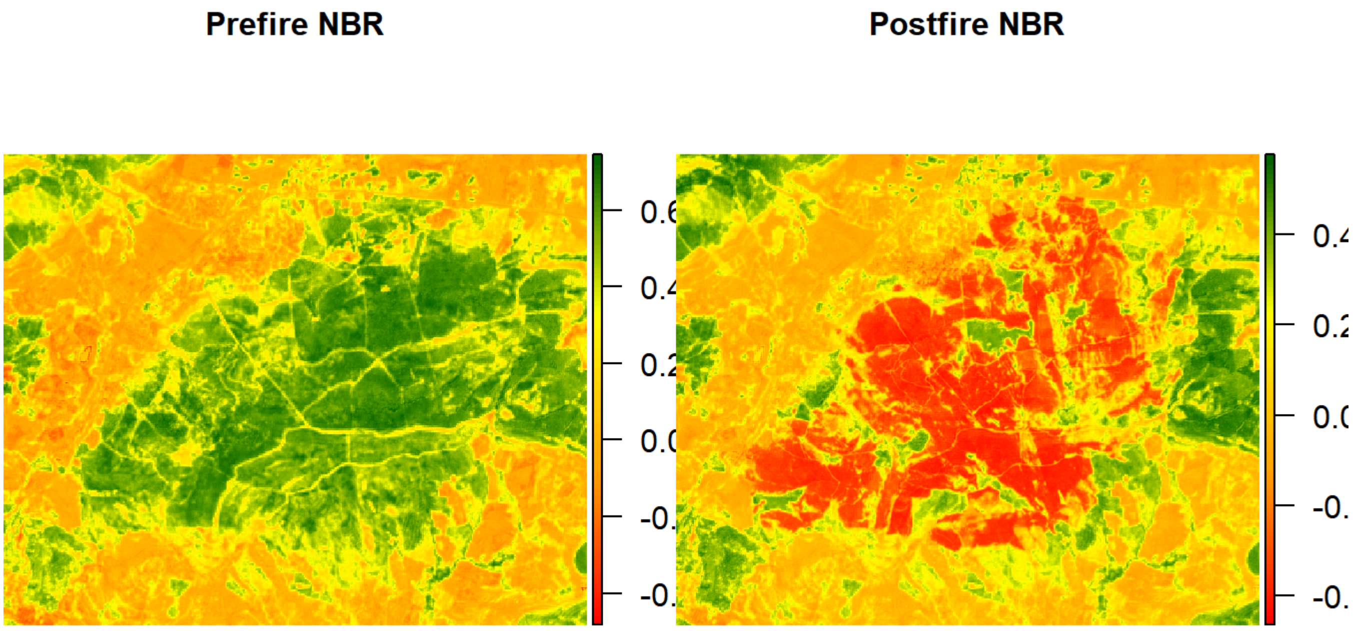
```r
# Step-by-Step Guide to Fire Burn Area Analysis Using Otsu's Thresholding Algorithm

# Load necessary libraries
library(raster)
```

```
## Loading required package: sp
```

```r
library(sf)
```

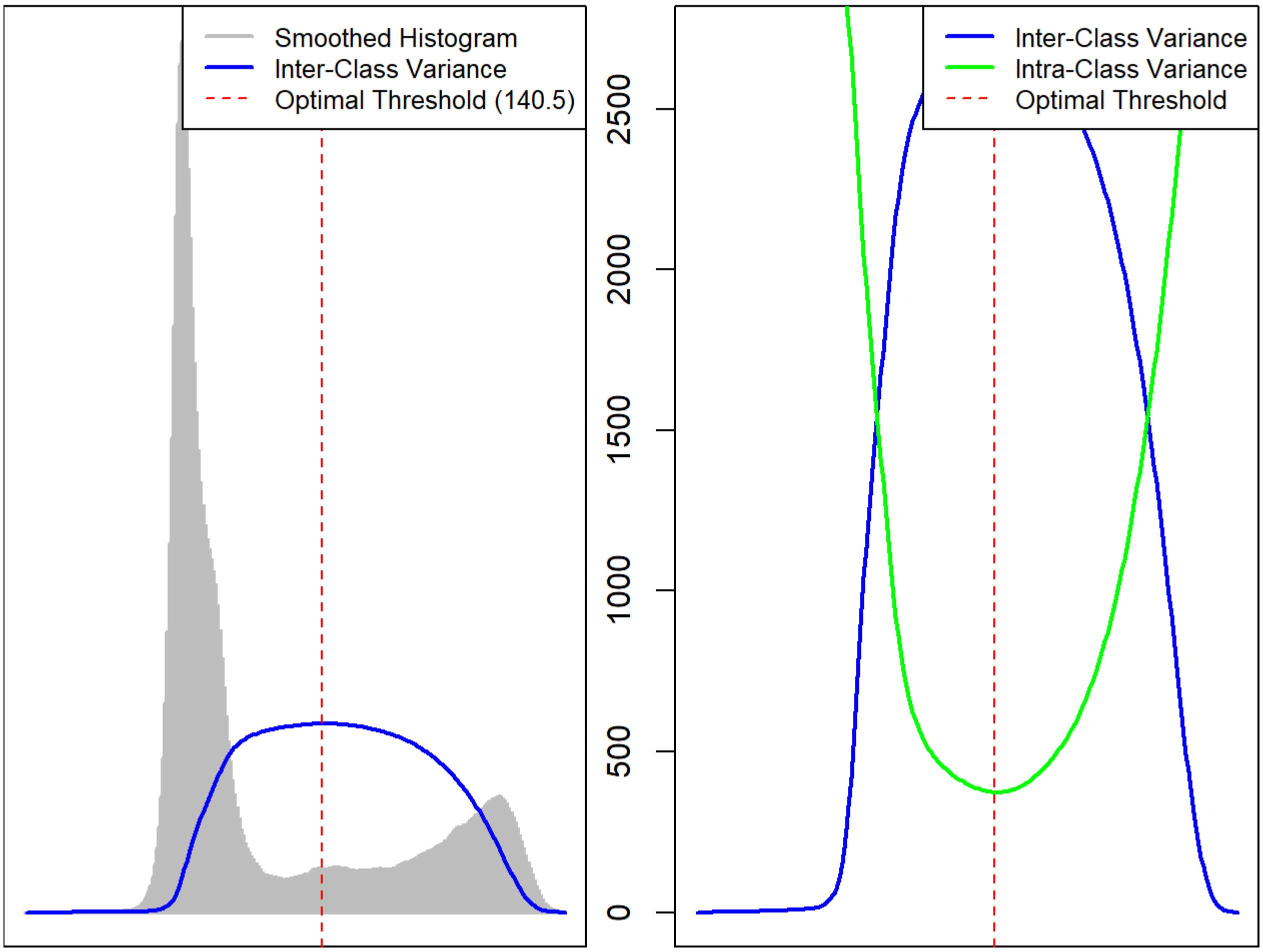
```
## Linking to GEOS 3.10.2, GDAL 3.4.1, PROJ 7.2.1; sf_use_s2() is TRUE
```

```r
# library(terra)
# Install the otsuSeg R package from a local source file (if not installed)
# install.packages("~/otsuSeg_0.1.0.tar.gz", repos = NULL, type = "source")

# Load the otsuSeg package
library(otsuSeg)

# Load the NBR (Normalized Burn Ratio) pre- and post-fire raster data from the otsuSeg package
NBRpre <- get_external_data("NBRpre")
NBRpost <- get_external_data("NBRpost")
NBRpre<-raster(NBRpre)
NBRpost<-raster(NBRpost)

# Set up the plotting parameters for side-by-side visualization
par(mfrow = c(1, 2), mar = c(0, 0, 0.5, 0.5))

# Define a custom color palette to highlight burned areas
burned_colors <- colorRampPalette(c("red", "orange", "yellow", "darkgreen"))(100)

# Plot NBR Pre-Fire
plot(NBRpre, col = burned_colors, axes = FALSE, box = FALSE, legend = TRUE)
title("Prefire NBR", line = -3, cex.main = 1)

# Plot NBR Post-Fire
plot(NBRpost, col = burned_colors, axes = FALSE, box = FALSE, legend = TRUE)
title("Postfire NBR", line = -3, cex.main = 1)
```



```
result <- binarize_raster(NBRpre, NBRpost, output_shapefile = FALSE)
```

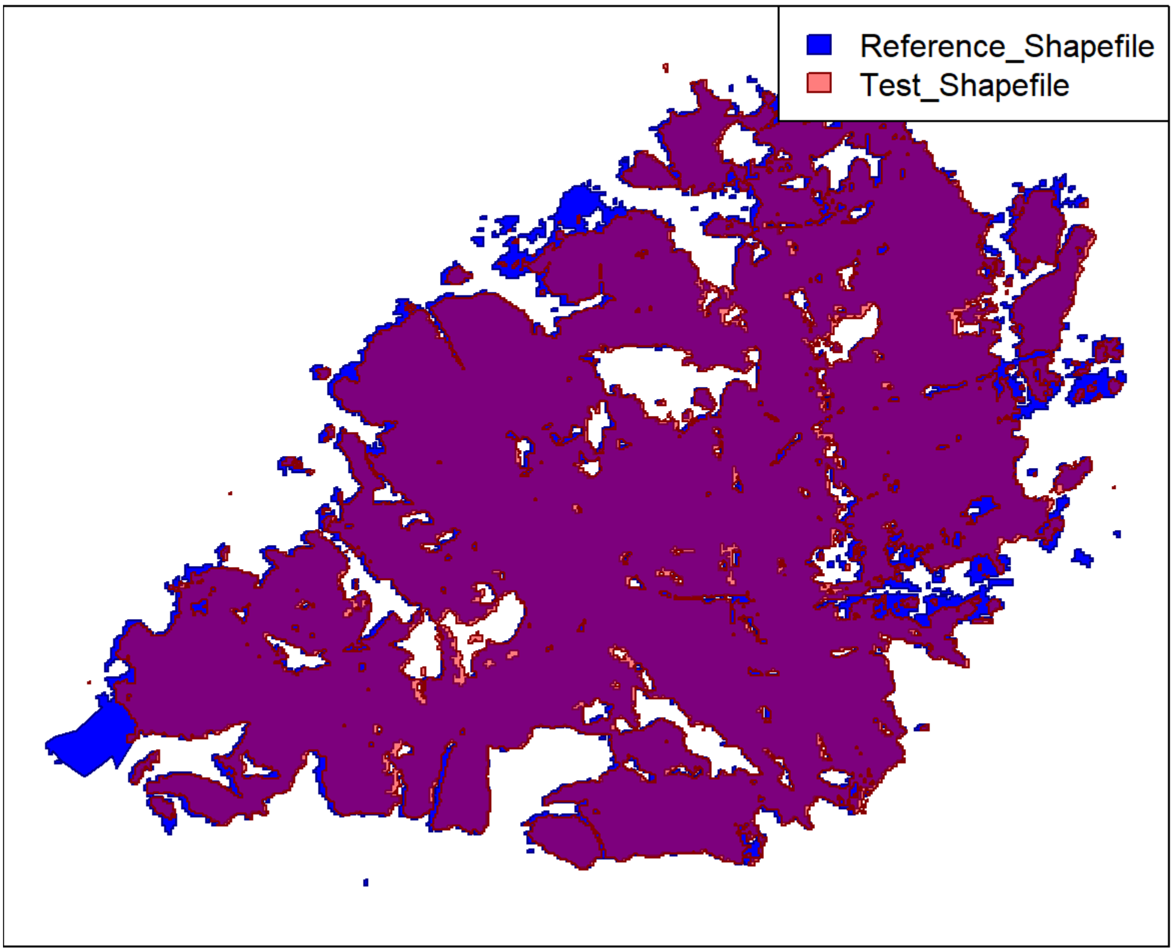
```
## Loading required namespace: rgeos
```



```
# If output_shapefile = TRUE, specify a path to save the binary raster as a shapefile:
# result <- binarize_raster(NBRpre, NBRpost, output_shapefile = TRUE, shapefile_path = "~/binary_raster.shp")

# Load the reference shapefile for quality control
shapefile_reference <- get_external_data("shapefile_reference")
shapefile_reference<- st_read(shapefile_reference)
```

```
## Reading layer `shapefile_reference' from data source
##   `C:\Users\achour\Documents\R\win-library\4.1\otsuSeg\extdata\shapefile_reference.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 1 feature and 1 field
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 510109.2 ymin: 4099395 xmax: 515554.2 ymax: 4103520
## Projected CRS: WGS 84 / UTM zone 32N
```

```
# Extract the generated binary shapefile
shapefile_test <- result$binary_shapefile
st_crs(shapefile_test) <- st_crs(shapefile_reference)

# Create a plot with both shapefiles

# Set up the plot window
plot(st_geometry(shapefile_reference), col = "blue", axes = FALSE,border = "darkblue", lwd = 1)

# Add the test shapefile to the plot with a different color
plot(st_geometry(shapefile_test), col = rgb(1, 0, 0, 0.5), add = TRUE, border = "darkred", lwd = 1)

# Optionally, add a legend
legend("topright", legend = c("Reference_Shapefile", "Test_Shapefile"),
       fill = c("blue", rgb(1, 0, 0, 0.5)), border = c("darkblue", "darkred"))
box()
```



```
# Perform quality control (QC) by comparing the generated shapefile with the reference
qc_results <- Quality_control(shapefile_test, shapefile_reference)
```

```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries

## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```

```
# Optionally, compute specific quality metrics, such as Similarity Size ("SimSize")
# qc_results <- Quality_control(shapefile_test, shapefile_reference, metrics = c("SimSize"))

# Print the QC results
print(qc_results)
```

```
##       Metric       Value
## 1  Precision  0.96091269
## 2     Recall  0.89824448
## 3   F1_Score  0.92852238
## 4        IoU  0.86658121
## 5         OS  0.10175552
## 6         US  0.03908731
## 7          E  0.07147762
## 8    SimSize  0.93478262
## 9        Loc 22.36760840
## 10       AFI  0.86658121
```