Step 2.2 Masking rasters

Diane ESPEL

2025-10-31

Contents

1 Objectives

This script allows multiple rasters to be cropped using a common vector layer referred to as the vector_mask. In this case, the goal is to mask a multispectral raster, a Digital Elevation Model (DEM), and a slope raster. To streamline the process, a custom masking function (`RasterMasking()`) is defined.

When masking a raster stack, it's crucial to ensure that all bands within the stack share the same spatial resolution. To achieve this, a specific function (`is_Same_Resolution()`) is also defined

2 Script explanation

2.1 Clean environment and graphics

```r
rm(list = ls()) # Clear all objects from the R environment to start fresh
graphics.off()   # Close all graphics devices (if any plots are open)
```

2.2 Load required packages

```r
library(sp)        # For spatial data compatibility
library(sf)        # For reading and manipulating shapefiles
library(terra)     # For raster and spatial operations
```

2.3 Create functions

We need to declare two functions:

- **is_Same_Resolution()** function that takes two raster objects (band1 and band2) and check if their spatial resolutions (i.e., pixel sizes) are identical. This is useful to ensure compatibility before performing operations that require matched resolutions.

  Parameters:

  - `band1`: The first raster band (e.g., blue band). -`band2`: The second raster band (e.g., NIR band).

  Returns:

  - `TRUE` if the resolutions are the same, otherwise `FALSE`."

- **Raster_Masking()** function that masks input raster (MS, Slope, or DTM) using a polygon mask and performs resolution adjustments for MS rasters. It supports MS rasters (with resolution correction) and mono-band rasters (slope or DTM). Before applying the mask, the function performs key checks to ensure consistency and compatibility:

- Verifies that RGB and NIR bands have identical spatial resolution with the`is_Same_Resolution()` function

- Confirms that all raster and vector layers use the same Coordinate Reference System (CRS)

- Applies the Region of Interest (ROI) mask to isolate relevant areas (e.g., islands)

```r
# Function to check if two rasters have the same resolution
is_Same_Resolution <- function(band1, band2){
    # Extract the resolution (pixel size) of each raster
    res_band1 <- res(band1)
    res_band2 <- res(band2)
    # Compare the resolutions and return TRUE if they are equal, otherwise FALSE
    return(all.equal(res_band1, res_band2) == TRUE)
  }

# Function to mask raster based on polygon and handle MS-specific issues
Raster_Masking <- function(archipelago, island, vector_mask, raster, nameSat, year, month = NULL,
↪  resolution, raster_type, save_path){

  # Reproject vector if CRS differ
  if (!crs(raster) == crs(vector_mask)) {
    vector_mask <- project(vector_mask, crs(raster))
  }

  num_bands <- nlyr(raster)

  if (raster_type == "PMS") {
```

```
    # Harmonize resolution if needed
    res_ok <- is_Same_Resolution(raster[[1]], raster[[num_bands]])
    if (!res_ok) {
      factor <- res(raster[[1]])[1] / res(raster[[num_bands]])[1]
      raster[[num_bands]] <- aggregate(raster[[num_bands]], fact = factor, fun = mean)
    }

    # Mask each band individually and save
    for (i in 1:num_bands) {
      path_band <- file.path(save_path, paste0(archipelago, "_", island, "_", nameSat, "_", year,
↪ "_", month, "_", resolution, "_band", i, "_cut.TIF"))
      mask(raster[[i]], vector_mask, filename = path_band, overwrite = TRUE)
    }

  } else {
    # For slope or DTM
    path_raster <- file.path(save_path, paste0(archipelago, "_", island, "_", nameSat, "_", year,
↪ "_", resolution, "_", raster_type, "_cut.TIF"))
    mask(raster, vector_mask, filename = path_raster, overwrite = TRUE)
  }
}
```

2.4 Define global variables

Note: A global variable is a variable defined outside of any function. This means the variable is accessible from any part of the code, including inside functions. A global variable retains its value throughout the execution of the R script unless it is explicitly modified in the code.

It is important to define at a minimum:

- the "District": the archipelago of interest (e.g. "CRO" for Crozet archipelago)
- the "Island": the island within the archipelago of interest (e.g. "POS" for Possession island)
- the "Satellite1": the name of satellite providing multispectral imagery
- the "Satellite2": the name of satellite providing terrain imagery
- the "Year2": the acquisition year of terrain imagery
- the "Res1": the resolution of the multispectral imagery
- the "Res2": the resolution of terrain imagery

```
District='CRO' # 3-letter code for archipelago (e.g. Crozet)
Island='POS'   # 3-letter code for island (e.g. Possession)
Satellite1="Pleiades" # satellite name of multispectral imagery
Res1 = "50cm"   # spatial resolution of multispectral imagery
Satellite2='SRTM'   # satellite name of DEM
Year2="2012"   # acquisition year of DEM
Res2="30m" # spatial resolution of DEM
```

2.5 Set working directory

To optimize memory, you must define one general root directory ("localHOME") that serves as the base path for your input and output data, respectively. This directory should point to the local environment where:

- input rasters are located under "data/raster/Precut_image"
- input ROI mask is located under "data/vector/mask"

- outputs cut rasters will be saved under `"data/raster/Cut_image"`

```r
# Base local path (customize to your local environment)
localHOME=paste0("/home/genouest/cnrs_umr6553/despel/CARTOVEGE/")
#localHOME = paste0("your_local_path/")

# path where to open vector data
open_mask_path=paste0(localHOME,"data/vector/mask")

# path where to open raster data
open_precut_raster_path=paste0(localHOME,"data/raster/Precut_image")

# path where to save your results
save_cut_raster_path=paste0(localHOME,"data/raster/Cut_image")
```

2.6 Load rasters and mask

```r
all_years <- c("2025","2024","2023","2022","2021","2020","2015","2011")
all_months <- c("01","02","03","04","05","06","07","08","09","10","11","12")

# Load ROI polygon shapefile
print("Loading island ROI shapefile")
ROI_mask <- st_read(dsn = paste0(open_mask_path, "/", District, "_", Island, "_POLY_", Year1,
↪  "_EPSG32743.shp"))
ROI_mask<- vect(ROI_mask) # convert to SpatVector

print("Loading slope raster")
raster_slope <- rast(paste0(open_precut_raster_path, "/", District, "_", Island, "_", Satellite2,
↪  "_", Year2, "_", Res2, "_slope_precut.tif"))

print("Loading DTM raster")
raster_DTM <- rast(paste0(open_precut_raster_path, "/", District, "_", Island, "_", Satellite2,
↪  "_", Year2, "_", Res2, "_dtm_precut.tif"))
```

2.7 Apply RasterMasking functions on MS rasters

Apply the `Raster_Masking()` function on the corresponding rasters when years and months exist

```r
for (Year1 in all_years) {

  print(paste0("Year: ", Year1))

  for (Month1 in all_months) {
    print(paste0("Month: ", Month1))

    raster_path <- file.path(open_precut_raster_path, paste0(District, "_", Island, "_",
↪  Satellite1, "_", Year1, "_", Month1, "_", Res1, "_PMS_precut.tif"))

    if (file.exists(raster_path)) {

      print("Loading MS raster")
      raster_MS <- rast(raster_path)

      print("Masking MS raster using ROI polygon")
```

```
    Raster_Masking(archipelago = District, island = Island, vector_mask = ROI_mask, raster =
    ↪  raster_MS, nameSat = Satellite1, year = Year1, month = Month1, resolution = Res1,
    ↪  raster_type = "PMS",save_path = save_cut_raster_path)
  } else {
    message("Raster not found for Year ", Year1, " Month ", Month1, " -> skipping.")
  }
 }
}
```

2.8   Apply RasterMasking functions on DEM rasters

Apply the `Raster_Masking()` function on the DEM corresponding rasters

```
# Mask slope raster
print("Masking slope raster using ROI polygon")
Raster_Masking(archipelago = District, island = Island, vector_mask = ROI_mask, raster =
↪  raster_slope, nameSat = Satellite2, year = Year2, resolution = Res2, raster_type =
↪  "slope",save_path = save_cut_raster_path)

# Mask DTM raster
print("Masking DTM raster using ROI polygon")
Raster_Masking(archipelago = District, island = Island, vector_mask = ROI_mask, raster =
↪  raster_DTM, nameSat = Satellite2, year = Year2, resolution = Res2, raster_type =
↪  "dtm",save_path = save_cut_raster_path)
```

This script masks multispectral, slope, and DTM rasters using a polygon shapefile, ensuring resolution consistency and proper CRS alignment.