> #setwd('C://Users//rdelaram//Documents//GitHub//eride//data//')

> # Find a spacious dir

> setwd('G:/')

> require(rgrass)

> require(sf)

> require(rgrass)

> require(terra)

> require(sf)

> #setwd('C://Users//rdelaram//Documents//GitHub//eride//data//')

> # Find a spacious dir

> setwd('G:/')

> # Find grass on Windows

> grassDir='C:/Program Files/GRASS GIS 8.2'

> # Import and reproject pop for pop at risk (PAR) calculation

> pop <- 'G:/indonesia/idn\_ppp\_2020.tif' # 100 m

> rgrass::initGRASS(gisBase = grassDir,

+ SG = rast(pop),

+ gisDbase = "grassdb",

+ location = "voronoi",

+ mapset = "PERMANENT",

+ override = TRUE,

+ remove\_GISRC = TRUE)

gisdbase grassdb

location voronoi

mapset PERMANENT

rows 20506

columns 55248

north 6.077917

south -11.01042

west 94.97208

east 141.0121

nsres 0.0008333335

ewres 0.0008333337

projection +proj=longlat +datum=WGS84 +no\_defs +type=crs

**Warning message:**

**In system(syscmd, intern = intern, ignore.stderr = ignore.stderr, :**

**running command 'g.proj.exe -w' had status 884**

> rgrass::execGRASS("r.in.gdal",

+ input = pop,

+ output = "pop",

+ flags = c("overwrite"))

Importing raster map <pop>...

100%

> rgrass::execGRASS("g.list", type = "raster")

pop

> execGRASS("g.region", flags = "p", intern = TRUE)

[1] "projection: 3 (Latitude-Longitude)" "zone: 0"

[3] "datum: wgs84" "ellipsoid: wgs84"

[5] "north: 6:04:40.5012N" "south: 11:00:37.512S"

[7] "west: 94:58:19.488E" "east: 141:00:43.56E"

[9] "nsres: 0:00:03.000001" "ewres: 0:00:03.000001"

[11] "rows: 20506" "cols: 55248"

[13] "cells: 1132915488"

> # hotspot

> execGRASS("r.hotspot", input="pop", output="hotspot\_pop", statistic="Gi\*")

**Error in system(cmd0, intern = TRUE) : 'r.hotspot.exe' not found**

**Error in parseGRASS(cmd, legacyExec = legacyExec) : r.hotspot not found**

> # Step 1: Calculate 90th percentile

> execGRASS("r.quantile", input="your\_raster", percentiles="90", output="quantile\_90.txt")

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**r.quantile**

**Invalid parameter name: outputCompute quantiles using two passes.**

**Invalid parameter name: outputraster, álgebra, estatisticas, percentile, quantile**

**Invalid parameter name: outputlist(c(name = "input", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster de entrada", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "quantiles", type = "integer", required = "no", multiple = "no", desc = "Número de quantis", default = "4", keydesc\_count = NA, keydesc = NA), c(name = "percentiles", type = "float", required = "no", multiple = "yes", desc = "List of percentiles", default = NA, keydesc\_count = NA, keydesc = NA), c(name = "bins", type = "integer",**

**required = "no", multiple = "no", desc = "Number of bins to use", default = "1000000", keydesc\_count = NA, keydesc = NA), c(name = "file", type = "string", required = "no", multiple = "no", desc = "Nome para o arquivo de saída (se omitido ou \"-\" saída para stdout)", default = NA,**

> # Step 1: Calculate 90th percentile

> execGRASS("r.quantile", input="pop", percentiles="90", output="quantile\_90.txt")

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**r.quantile**

**Invalid parameter name: outputCompute quantiles using two passes.**

**Invalid parameter name: outputraster, álgebra, estatisticas, percentile, quantile**

**Invalid parameter name: outputlist(c(name = "input", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster de entrada", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "quantiles", type = "integer", required = "no", multiple = "no", desc = "Número de quantis", default = "4", keydesc\_count = NA, keydesc = NA), c(name = "percentiles", type = "float", required = "no", multiple = "yes", desc = "List of percentiles", default = NA, keydesc\_count = NA, keydesc = NA), c(name = "bins", type = "integer",**

**required = "no", multiple = "no", desc = "Number of bins to use", default = "1000000", keydesc\_count = NA, keydesc = NA), c(name = "file", type = "string", required = "no", multiple = "no", desc = "Nome para o arquivo de saída (se omitido ou \"-\" saída para stdout)", default = NA,**

> # Step 1: Calculate 90th percentile

> execGRASS("r.quantile", input="pop", percentiles="90", file="quantile\_90.txt")

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**Parameter <percentiles> does not have numeric value**

> execGRASS("r.quantile", input="pop", percentiles=90, file="quantile\_90.txt")

Computing histogram

100%

Computing bins

Binning data

100%

Sorting bins

Computing quantiles

> # Step 2: Read the 90th percentile value

> percentile\_value <- as.numeric(readLines("quantile\_90.txt")[1])

**Warning message:**

**NAs introduced by coercion**

> percentile\_value

[1] NA

>

> quantile\_contents <- readLines("quantile\_90.txt")

> quantile\_contents

[1] "0:90.000000:2.056477"

> # Read the contents of quantile\_90.txt

> quantile\_contents <- readLines("quantile\_90.txt")

> # Split the first line by the colon (:) separator

> split\_values <- strsplit(quantile\_contents[1], ":")[[1]]

> # Extract the second value (90th percentile)

> percentile\_value <- as.numeric(split\_values[3]) # The 3rd element is the value

> # Check the extracted percentile value

> print(percentile\_value)

[1] 2.056477

> # Step 1: Calculate 90th percentile

> execGRASS("r.quantile", input="pop", percentiles=95, file="quantile\_95.txt")

Computing histogram

100%

Computing bins

Binning data

100%

Sorting bins

Computing quantiles

> # Read the contents of quantile\_90.txt

> quantile\_contents <- readLines("quantile\_95.txt")

> # Split the first line by the colon (:) separator

> split\_values <- strsplit(quantile\_contents[1], ":")[[1]]

> # Extract the second value (90th percentile)

> percentile\_value <- as.numeric(split\_values[3]) # The 3rd element is the value

> # Check the extracted percentile value

> print(percentile\_value)

[1] 4.867154

> print(percentile\_value)

[1] 4.867154

> # Step 3: Reclassify the raster to keep pixels above the 90th percentile

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules="-", stdin=c(

+ paste0(percentile\_value, " thru max = 1"),

+ "\* = NULL" ))

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**r.reclass**

**Invalid parameter name: stdinReclassify raster map based on category values. Creates a new raster map whose category values are based upon a reclassification of the categories in an existing raster map.**

**Invalid parameter name: stdinraster, reclassificação**

**Invalid parameter name: stdinlist(c(name = "input", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster a ser reclassificado", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "output", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster de saída", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "rules", type = "string", required = "yes", multiple = "no", desc = "'-' para entrada padrão", default = NA, keydesc\_count = "1", keydesc = "name"**

**), c(name = "title", type = "string", required = "no", multiple = "no", desc = "Título para mapa raster de saída", default = NA, keydesc\_count = NA, keydesc = NA))**

**Invalid parameter name**

> percentile\_value

[1] 4.867154

> print(percentile\_value)

[1] 4.867154

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules="-",

+ stdin=c(

+ paste0(percentile\_value, " thru max = 1"), # Values above 95th percentile

+ "\* = NULL" # Everything else is NULL

+ ))

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**r.reclass**

**Invalid parameter name: stdinReclassify raster map based on category values. Creates a new raster map whose category values are based upon a reclassification of the categories in an existing raster map.**

**Invalid parameter name: stdinraster, reclassificação**

**Invalid parameter name: stdinlist(c(name = "input", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster a ser reclassificado", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "output", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster de saída", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "rules", type = "string", required = "yes", multiple = "no", desc = "'-' para entrada padrão", default = NA, keydesc\_count = "1", keydesc = "name"**

**), c(name = "title", type = "string", required = "no", multiple = "no", desc = "Título para mapa raster de saída", default = NA, keydesc\_count = NA, keydesc = NA))**

**Invalid parameter name**

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules="-",

+ stdin=c(

+ paste0(percentile\_value, " thru max = 1"), # Values above 95th percentile

+ "\* = NULL" ))

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**r.reclass**

**Invalid parameter name: stdinReclassify raster map based on category values. Creates a new raster map whose category values are based upon a reclassification of the categories in an existing raster map.**

**Invalid parameter name: stdinraster, reclassificação**

**Invalid parameter name: stdinlist(c(name = "input", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster a ser reclassificado", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "output", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster de saída", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "rules", type = "string", required = "yes", multiple = "no", desc = "'-' para entrada padrão", default = NA, keydesc\_count = "1", keydesc = "name"**

**), c(name = "title", type = "string", required = "no", multiple = "no", desc = "Título para mapa raster de saída", default = NA, keydesc\_count = NA, keydesc = NA))**

**Invalid parameter name**

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules="-",

+ stdin=c(paste0(percentile\_value, " thru max = 1"), # Values above 95th percentile

+ "\* = NULL" ))

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**r.reclass**

**Invalid parameter name: stdinReclassify raster map based on category values. Creates a new raster map whose category values are based upon a reclassification of the categories in an existing raster map.**

**Invalid parameter name: stdinraster, reclassificação**

**Invalid parameter name: stdinlist(c(name = "input", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster a ser reclassificado", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "output", type = "string", required = "yes", multiple = "no", desc = "Nome do mapa raster de saída", default = NA, keydesc\_count = "1", keydesc = "name"), c(name = "rules", type = "string", required = "yes", multiple = "no", desc = "'-' para entrada padrão", default = NA, keydesc\_count = "1", keydesc = "name"**

**), c(name = "title", type = "string", required = "no", multiple = "no", desc = "Título para mapa raster de saída", default = NA, keydesc\_count = NA, keydesc = NA))**

**Invalid parameter name**

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # Values above 95th percentile

+ "\* = NULL" # Everything else is NULL

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # Execute r.reclass using the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

5.367154 rounded up to 5

ERRO:4.867154 thru max = 1 - invalid reclass rule

> reclass\_rules

[1] "4.867154 thru max = 1" "\* = NULL"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

5.367154 rounded up to 5

ERRO:4.867154 thru max = 1 - invalid reclass rule

> # Clean up the temporary file if needed

> unlink(rules\_file)

> # Step 1: Convert the 90th percentile raster to points

> execGRASS("r.to.vect", input="above\_95th\_percentile", output="percentile\_points", type="point")

ERRO:Mapa raster <above\_95th\_percentile> n�o encontrado

> # Step 1: Convert the 90th percentile raster to points

> execGRASS("r.to.vect", input="above\_95th\_percentile", output="percentile\_points", type="point")

ERRO:Mapa raster <above\_95th\_percentile> n�o encontrado

> rgrass::execGRASS("g.list", type = "raster")

pop

> print(readLines(rules\_file))

**Error in file(con, "r") : cannot open the connection**

**In addition: Warning message:**

**In file(con, "r") :**

**cannot open file 'C:\Users\rdelaram\AppData\Local\Temp\RtmpwPZS8Y\file1a703dd52620.txt': No such file or directory**

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # Values above 95th percentile

+ "\* = NULL" # Everything else is NULL

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

5.367154 rounded up to 5

ERRO:4.867154 thru max = 1 - invalid reclass rule

> percentile\_value <- round(percentile\_value, 3)

> percentile\_value

[1] 4.867

> percentile\_value <- round(percentile\_value)

> percentile\_value

[1] 5

> # sharp round it

> percentile\_value <- round(percentile\_value)

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # Values above 95th percentile

+ "\* = NULL" # Everything else is NULL

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> rules\_file

[1] "C:\\Users\\rdelaram\\AppData\\Local\\Temp\\RtmpwPZS8Y\\file1a705a0823e2.txt"

> rules\_file

[1] "C:\\Users\\rdelaram\\AppData\\Local\\Temp\\RtmpwPZS8Y\\file1a705a0823e2.txt"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 - invalid reclass rule

> reclass\_rules <- c(

+ paste0((percentile\_value + 1e-6), " thru max = 1"), # Values strictly greater than the 95th percentile

+ "\* = NULL"

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

5.500001 rounded up to 5

ERRO:5.000001 thru max = 1 - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # Values greater than or equal to the 95th percentile

+ "\* = NULL" # Everything else is NULL

+ )

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

5.500001 rounded up to 5

ERRO:5.000001 thru max = 1 - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # Values greater than or equal to the 95th percentile

+ "\* = NULL" # Everything else is NULL

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, ":\*", " = 1"), # Values greater than or equal to 95th percentile become 1

+ "\* = NULL" # All other values become NULL

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5:\* = 1 - invalid reclass rule

> reclass\_rules

[1] "5:\* = 1" "\* = NULL"

> reclass\_rules

[1] "5:\* = 1" "\* = NULL"

> rules\_file

[1] "C:\\Users\\rdelaram\\AppData\\Local\\Temp\\RtmpwPZS8Y\\file1a707772755a.txt"

> reclass\_rules

[1] "5:\* = 1" "\* = NULL"

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru \* = 1 High\_Values"), # Values above the 95th percentile

+ "\* = 0 Low\_Values" # All other values become 0

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru \* = 1 High\_Values - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru \* = 1 high"), # Values above the 95th percentile

+ "\* = 0" # All other values become 0

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru \* = 1 high - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru \* = 1 high"), # Values above the 95th percentile

+ "\* = 0" # All other values become 0

+ )

> reclass\_rules

[1] "5 thru \* = 1 high" "\* = 0"

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru \* = 1 high"), # Values above the 95th percentile get 1 and labeled as "forest"

+ "\* = 0" # Everything else becomes 0

+ )

> reclass\_rules

[1] "5 thru \* = 1 high" "\* = 0"

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru \* = 1 high"), # Values above the 95th percentile get 1 and labeled as "forest"

+ "\* = 0" # Everything else becomes 0

+ )

> reclass\_rules

[1] "5 thru \* = 1 high" "\* = 0"

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru \* = 1 high - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 forest"), # 95th percentile and above gets reclassified to 1

+ "\* = 0" # All other values become 0

+ )

> # Write the rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(reclass\_rules, con = rules\_file)

> rules\_file

[1] "C:\\Users\\rdelaram\\AppData\\Local\\Temp\\RtmpwPZS8Y\\file1a7047a64e8c.txt"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 forest - invalid reclass rule

> rules\_file

[1] "C:\\Users\\rdelaram\\AppData\\Local\\Temp\\RtmpwPZS8Y\\file1a7047a64e8c.txt"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=reclass\_rules)

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**Parameter <rules> has multiple values**

> reclass\_rules

[1] "5 thru max = 1 forest" "\* = 0"

> # Reclass rules in the correct format using 'max' instead of '\*'

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 forest"), # 95th percentile and above gets reclassified to 1

+ "\* = 0" # All other values become 0

+ )

> reclass\_rules

[1] "5 thru max = 1 forest" "\* = 0"

> # Concatenate rules into a single string

> rules\_string <- paste(reclass\_rules, collapse = "\n")

> rules\_string

[1] "5 thru max = 1 forest\n\* = 0"

> # Write the reclass rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(rules\_string, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=reclass\_rules)

**Error in doGRASS(cmd, flags = flags, ..., parameters = parameters, echoCmd = echoCmd, :**

**Parameter <rules> has multiple values**

> reclass\_rules

[1] "5 thru max = 1 forest" "\* = 0"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_string)

ERRO:Cannot open rules file <5 thru max = 1 forest

\* = 0>

> rules\_string

[1] "5 thru max = 1 forest\n\* = 0"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_string)

ERRO:Cannot open rules file <5 thru max = 1 forest

\* = 0>

> rules\_string

[1] "5 thru max = 1 forest\n\* = 0"

> # Write the reclass rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(rules\_string, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 forest - invalid reclass rule

> # Reclass rules in the correct format using 'max' instead of '\*'

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # 95th percentile and above gets reclassified to 1

+ "\* = 0" # All other values become 0

+ )

> # Concatenate rules into a single string

> rules\_string <- paste(reclass\_rules, collapse = "\n")

> # Write the reclass rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(rules\_string, con = rules\_file)

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 - invalid reclass rule

>

>

> rules\_string <- paste0(percentile\_value, " thru max = 1 forest\n\* = 0")

>

> # Execute r.reclass with the defined rules

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_string)

ERRO:Cannot open rules file <5 thru max = 1 forest

\* = 0>

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 forest"), # 95th percentile and above gets reclassified to 1

+ "\* = 0" # All other values become 0

+ )

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 high"), # 95th percentile and above gets reclassified to 1

+ "\* = 0" # All other values become 0

+ )

> # Concatenate rules into a single string for writing to file

> rules\_string <- paste(reclass\_rules, collapse = "\n")

> # Write the reclass rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(rules\_string, con = rules\_file)

> rules\_file

[1] "C:\\Users\\rdelaram\\AppData\\Local\\Temp\\RtmpwPZS8Y\\file1a70310e1546.txt"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 high - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 forest"), # Reclassify values above the 95th percentile

+ "\* = 0" # All other values become 0

+ )

>

> # Concatenate rules into a single string for writing to file

> rules\_string <- paste(reclass\_rules, collapse = "\n")

>

> # Write the reclass rules to a temporary file

> rules\_file <- tempfile(fileext = ".txt")

> writeLines(rules\_string, con = rules\_file)

>

> # Check the contents of the rules file for debugging

> cat("Contents of the rules file:\n")

Contents of the rules file:

> cat(readLines(rules\_file), sep = "\n")

5 thru max = 1 forest

\* = 0

>

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 forest - invalid reclass rule

>

> # Clean up the temporary file

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 forest"), # Reclassify values above the 95th percentile

+ "\* = 0" # All other values become 0

+ )

> reclass\_rules

[1] "5 thru max = 1 forest" "\* = 0"

> getwd()

[1] "G:/"

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1 forest"), # Reclassify values above the 95th percentile

+ "\* = 0" # All other values become 0

+ )

> # Specify the path for the rules file

> rules\_file <- "reclass\_rules.txt" # Change this to your desired file path

> # Write the reclass rules to a file

> writeLines(reclass\_rules, con = rules\_file)

> # Output the contents of the rules file to the console for verification

> cat("Contents of the rules file:\n")

Contents of the rules file:

> cat(readLines(rules\_file), sep = "\n")

5 thru max = 1 forest

\* = 0

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru max = 1 forest - invalid reclass rule

> reclass\_rules <- c(

+ paste0(percentile\_value, " thru max = 1"), # Reclassify values above the 95th percentile

+ "\* = 0" # All other values become 0

+ )

> # Specify the path for the rules file

> rules\_file <- "reclass\_rules.txt" # Change this to your desired file path

> # Write the reclass rules to a file

> writeLines(reclass\_rules, con = rules\_file)

> execGRASS("r.info", map="pop")

+----------------------------------------------------------------------------+

| Map: pop Date: Tue Oct 08 14:36:43 2024 |

| Mapset: PERMANENT Login of Creator: rdelaram |

| Location: voronoi |

| DataBase: grassdb |

| Title: |

| Timestamp: none |

|----------------------------------------------------------------------------|

| |

| Type of Map: raster Number of Categories: 0 |

| Data Type: FCELL Semantic label: (none) |

| Rows: 20506 |

| Columns: 55248 |

| Total Cells: 1132915488 |

| Projection: Latitude-Longitude |

| N: 6:04:40.500638N S: 11:00:37.499116S Res: 0:00:03 |

| E: 141:00:43.498022E W: 94:58:19.498685E Res: 0:00:03 |

| Range of data: min = 0 max = 325.1095 |

| |

| Data Description: |

| gerado por RINGDA~1 |

| |

| Comments: |

| RINGDA~1 --overwrite input="G:/indonesia/idn\_ppp\_2020.tif" output="p\ |

| op" memory=300 offset=0 num\_digits=0 |

| |

+----------------------------------------------------------------------------+

> execGRASS("r.info", map="pop")

+----------------------------------------------------------------------------+

| Map: pop Date: Tue Oct 08 14:36:43 2024 |

| Mapset: PERMANENT Login of Creator: rdelaram |

| Location: voronoi |

| DataBase: grassdb |

| Title: |

| Timestamp: none |

|----------------------------------------------------------------------------|

| |

| Type of Map: raster Number of Categories: 0 |

| Data Type: FCELL Semantic label: (none) |

| Rows: 20506 |

| Columns: 55248 |

| Total Cells: 1132915488 |

| Projection: Latitude-Longitude |

| N: 6:04:40.500638N S: 11:00:37.499116S Res: 0:00:03 |

| E: 141:00:43.498022E W: 94:58:19.498685E Res: 0:00:03 |

| Range of data: min = 0 max = 325.1095 |

| |

| Data Description: |

| gerado por RINGDA~1 |

| |

| Comments: |

| RINGDA~1 --overwrite input="G:/indonesia/idn\_ppp\_2020.tif" output="p\ |

| op" memory=300 offset=0 num\_digits=0 |

| |

+----------------------------------------------------------------------------+

>

>

>

> reclass\_rules <- c(

+ "0 thru max = 1" # All values become 1

+ )

>

> # Specify the path for the rules file

> rules\_file <- "simple\_reclass\_rules.txt"

>

> # Write the reclass rules to the file

> writeLines(reclass\_rules, con = rules\_file)

>

> # Execute the reclassification command

> execGRASS("r.reclass", input="pop", output="all\_one\_raster", rules=rules\_file)

ERRO:0 thru max = 1 - invalid reclass rule

>

> # Check the output raster

> execGRASS("r.info", map="all\_one\_raster")

ERRO:Mapa raster <all\_one\_raster> n�o encontrado

>

> rules\_string <- "0 = 1" # All zeros become 1

>

> # Execute the reclassification command directly with rules as a string

> execGRASS("r.reclass", input="pop", output="all\_one\_raster", rules=rules\_string)

ERRO:Cannot open rules file <0 = 1>

>

> # Check the output raster

> execGRASS("r.info", map="all\_one\_raster")

ERRO:Mapa raster <all\_one\_raster> n�o encontrado

> # Specify the path for the rules file

> rules\_file <- "my\_reclass\_rules.txt" # Change this to your desired file path

> rules\_file

[1] "my\_reclass\_rules.txt"

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:Cannot open rules file <my\_reclass\_rules.txt>

> rules\_file

[1] "my\_reclass\_rules.txt"

> getwd()

[1] "G:/"

> # Specify the path for the rules file

> rules\_file <- "my\_reclass\_rules.txt" # Change this to your desired file path

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:Cannot open rules file <my\_reclass\_rules.txt>

> rules\_file\_path <- file.path(getwd(), "my\_reclass\_rules.txt")

> rules\_file\_path

[1] "G://my\_reclass\_rules.txt"

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file\_path)

ERRO:Cannot open rules file <G://my\_reclass\_rules.txt>

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules='5 thru max = 1 high

+ \* = 0')

ERRO:Cannot open rules file <5 thru max = 1 high

\* = 0>

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file\_path)

ERRO:Cannot open rules file <G://my\_reclass\_rules.txt>

> rules\_file\_path

[1] "G://my\_reclass\_rules.txt"

> # Specify the path for the rules file

> rules\_file <- "my\_reclass\_rule.txt" # Change this to your desired file path

> rules\_file\_path <- file.path(getwd(), "my\_reclass\_rules.txt")

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file\_path)

ERRO:Cannot open rules file <G://my\_reclass\_rules.txt>

> # Specify the path for the rules file

> rules\_file <- "my\_reclass\_rule.txt" # Change this to your desired file path

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

2.501000 rounded up to 2

6.501000 rounded up to 6

12.501000 rounded up to 12

> # Specify the path for the rules file

> rules\_file <- "my\_reclass\_rule.txt" # Change this to your desired file path

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:op��o <output>: <above\_95th\_percentile> existe. Para substituir, use a flag --overwrite

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file, flags = c("overwrite"))

ERRO:0 thru max = 1 - invalid reclass rule

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:op��o <output>: <above\_95th\_percentile> existe. Para substituir, use a flag --overwrite

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file,

+ flags = c("overwrite"))

ERRO:5 thru inf = 1 - invalid reclass rule

> rules\_file

[1] "my\_reclass\_rule.txt"

> # r.reclass rounds it automatically

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:op��o <output>: <above\_95th\_percentile> existe. Para substituir, use a flag --overwrite

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file,

+ flags = "overwrite")

ERRO:5 thru \* = 1 high - invalid reclass rule

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:op��o <output>: <above\_95th\_percentile> existe. Para substituir, use a flag --overwrite

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:op��o <output>: <above\_95th\_percentile> existe. Para substituir, use a flag --overwrite

> rgrass::execGRASS("g.list", type = "raster")

above\_95th\_percentile

pop

> execGRASS("g.remove", type="raster", name="above\_95th\_percentile", flags="f")

Removendo raster <above\_95th\_percentile>

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:5 thru \* = 1 high - invalid reclass rule

> # Execute r.reclass with the rules file

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

> rgrass::execGRASS("g.list", type = "raster")

above\_95th\_percentile

pop

> percentile\_value

[1] 5

> # Step 1: Convert the 90th percentile raster to points

> execGRASS("r.to.vect", input="above\_95th\_percentile", output="percentile\_points", type="point")

Driver padr�o / banco de dados definido para:

driver: sqlite

banco de dados: $GISDBASE/$LOCATION\_NAME/$MAPSET/sqlite/sqlite.db

Extracting points...

100%

Construindo topologia para mapa vetorial <percentile\_points@PERMANENT>...

Registrando primitivas...

Regi�o atual linhas: 20506, colunas: 55248

ERRO:G\_realloc: n�o foi poss�vel alocar 1352520000 bytes de mem�ria em

lib/vector/diglib/cindex.c:113

dbmi: Erro de protocolo

dbmi: Erro de protocolo

> system("r.report map=above\_95th\_percentile units=c,p")

[1] 127

> # Alternatively, you can use r.stats to get the count of pixels with value 1:

> pixel\_count <- execGRASS("r.stats", input = "above\_95th\_percentile", flags = c("n", "quiet"), intern = TRUE)

> pixel\_count

[1] "0" "1"

> pixel\_count\_1 <- pixel\_count[pixel\_count == "1"]

> print(pixel\_count\_1)

[1] "1"

> pixel\_count <- execGRASS("r.stats", input = "above\_95th\_percentile", flags = c("n", "quiet"), intern = TRUE)

>

> # Filter for the count of pixels where value is 1

> pixel\_count\_1 <- pixel\_count[grep("^1 ", pixel\_count)]

> print(pixel\_count\_1)

character(0)

>

> rules\_file

[1] "my\_reclass\_rule.txt"

> rules\_file

[1] "my\_reclass\_rule.txt"

> rules\_file\_path <- file.path(getwd(), "my\_reclass\_rule.txt")

> # (omg this was stressful!)

> execGRASS("r.reclass", input="pop", output="above\_95th\_percentile", rules=rules\_file)

ERRO:op��o <output>: <above\_95th\_percentile> existe. Para substituir, use a flag --overwrite