Grid Statistics

Tina Deenik

21/04/2021

LiDAR grid statistic calculation:

Be sure not to just loop through individual files using a loop function. This is bad practice because each file is loaded without a buffer, meaning outputs will be invalid or weak at the edges because of the absence of spatial context.

Things you must ensure: your files are all unique (no duplicates), files do not overlap, and you must set your output folder every time you call in a new catalog.

```
#library(lidR)
#library(future)
#library(raster)
#library(tools)

##Simple example:

#ctg < - readctg("file.location")

#opt_output_files(las) <- "output.file.location/{ORIGINALFILENAME}_zmax"       ##Note that {ORIG
INALFILENAME} keeps the original file name in the new output location.

#zmax <- grid_metrics(ctg, max(Z), 10)           ##Run with a resolution of 10.
```

Optimize computation

```
##LidR will automatically assign a buffer to the files, or you can manually set it with this fun
ction:

#opt_chunk_buffer(ctg) <- 10

##You can also manually set the processing chunk size to aid in processing time (a bit finicky t
o figure out if its better or worse)

#opt_chunk_size(ctg) <- 20

##Compute using only some points: optimal. faster and uses less memory. No intermediate object

#metrics = grid_metrics(las, ~mean(Z), 10, filter = ~ReturnNumber == 1)

##OR Compute using only some points: best. ~50% faster and uses ~10x less memory

#las = readLAS(LASfile, filter = "-keep_first")
#metrics = grid_metrics(las, ~mean(Z), 10)
```

A real example

```
#alldata <- readLAScatalog("P:/AserLab/Data/LiDAR_Unique") #input las catalogue

##If you have a large amount of data, you may want to set opt_stop_early so it continues to proc
ess despite errors:

#opt_stop_early(alldata) <- FALSE

#opt_chunk_size(alldata) <- 0

#plot(alldata,chunk=TRUE) #visualize data

#lascheck(alldata) #check to make sure the data is good

#summary(alldata) #check data

#opt_filter(alldata) <- "-drop_class 7 - drop_z_below_0" #drop flagged outliers

#opt_select(alldata) <- "zi" #saves pulling in unnecessary data

#opt_output_files(alldata) <- "P:/AserLab/Projects/OK_Wetlands/LiDAR_gridmets/Individual_files_1
0m/{ORIGINALFILENAME}_metsimple"

##parallel processing, 8 cores
#plan(multisession, workers = 8L)
#set.seed(12345678)
#future.seed = TRUE

##track progress:
#opt_progress(alldata) <- TRUE

##Define your own new metrics
#myMetrics = function(z, i)
#{
#  metrics = list(
#    zmean = mean(z),
#    zmax  = max(z),
#    zsd = sd(z),
#    imax = max(i),
#    imean = mean(i),
#    imin = min(i),
#    isd = sd(i)
# )
#  return(metrics)
#}

#RunALL = grid_metrics(alldata, ~myMetrics(Z, Intensity), 10)
```

A note about naming:

If you want to rename your files, you must do it manually:

```
##if you call in the file, remember it is saved as a raster brick:

#bricktest <- brick("P:/AserLab/Projects/OK_Wetlands/TEST_OUTPUT/bc_082e003_1_2_2_xyes_12_utm11_2019_1.tif")

#summary(bricktest)
#names(bricktest)

##this will plot each of the different rasters in order of which you calculated them in myMetrics above.

#plot(bricktest[[1]])
#plot(bricktest[[2]])
#plot(bricktest[[3]])
#plot(bricktest[[4]])
#plot(bricktest[[5]])

#names(bricktest) <- paste0(names(bricktest), c('_coef_var', '_zmean', '_zmax', '_zmin', '_zsd'))
```

Now we need to mosaic all the grid stats into one file:

```
#indat = list_files_with_exts("P:/AserLab/Projects/OK_Wetlands/LiDAR_gridmets/Individual_files_10m", "tif", full.names = TRUE)

#mosaic_rasters(
# gdalfile = indat,
#  gdalwarp_index = 1,
#  dst_dataset = 'P:/AserLab/Projects/OK_Wetlands/LiDAR_gridmets/AllFiles_gridmetrics_April26_10m.tif',
#  output.vrt = 'P:/AserLab/Projects/OK_Wetlands/LiDAR_gridmets/AllFiles_gridmetrics_April26_10m.vrt',
#  )

#viewmetrics <- brick("P:/AserLab/Projects/OK_Wetlands/LiDAR_gridmets/AllFiles_gridmetrics_April26_10m.vrt")

#plot(viewmetrics) #should create a new plot for each of the metrics (7)

#plot(viewmetrics[[1]]) #will just plot the first metric
```

If your code crashes and you're left part way through, you do not need to restart from the beginning. Instead, do this:

```
#It crashed at 71%
#alldata$processed <- FALSE
#alldata$processed[c(2452:3430)] <- TRUE
```

Other interesting grid statistics that I might run include:

```
##SD of ground

#las <- readLAS("P:/AserLab/Data/LiDAR_Unique/bc_082e002_4_4_4_xyes_12_utm11_2018.laz")

#metrics <- grid_metrics(las, ~sd(Z), 10, filter = ~Classification ==2)

##SD of all hits

#sd(Z)

##Then do a raster calc on these two to figure out the ratio of SD all hits to SD ground hits

##entropy: quantifies diversity & evenness of point cloud heights. by = 1 partitions point cloud
in 1 m tall horizontal slices. ranges from 0-1, with 1 being more evenly distributed points acro
ss the 1 m tall slices

#entro <- entropy(z, by = 1, zmax = NULL)

##grid density
#d <- grid_density(las, 10)
#plot(d)
```