

# R workflow for forest gaps and edges detection from ALS data

Jean-Matthieu Monnet

2021-08-03

---

This tutorial presents R code for forest gaps and edges detection from Airborne Laser Scanning (ALS) data. The workflow is based on functions from packages `lidaRtRee` and `lidR`.

Licence: GNU GPLv3 / Source page

Many thanks to Pascal Obst  tar for checking code and improvement suggestions.

## Study area and ALS data

The study area is a 200m x 200m forest located in the Jura mountain (France). The following code shows how to extract the ALS data from a folder containing normalized LAS files, and compute the Canopy Height Model (CHM) at 1 m resolution.

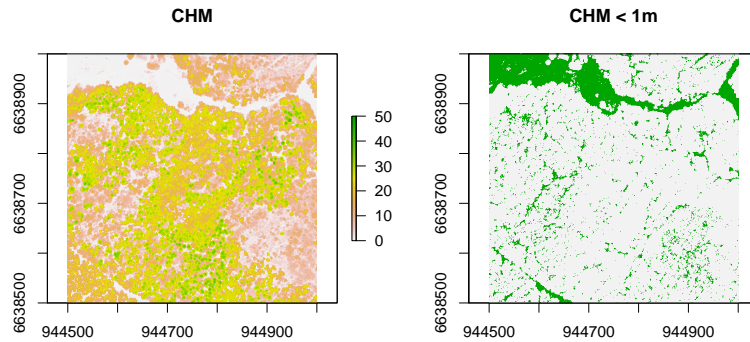
```
# build catalog from folder with normalized LAS/LAZ files
cata <- lidR::readALSLAScatalog("/las.folder/")
# set coordinate system
lidR::projection(cata) <- 2154
# define region of interest where points should be extracted
centre <- c(944750, 6638750)
size <- 250
# extract data on square area centered on centre
las <- lidR::clip_rectangle(cata, centre[1] - size, centre[2] - size, centre[1] +
  size, centre[2] + size)
# compute canopy height model from normalized point cloud
chm <- lidaRtRee::points2DSM(las, 1, centre[1] - size, centre[1] + size, centre[2] -
  size, centre[2] + size)
# save chm in Rdata file save(chm, file='chm.rda')
```

If the CHM has been previously calculated.

```
# load dataset
load(file = "../data/gap.detection/chm.rda")
```

Missing, low and high values are replaced for better visualization and processing.

```
# replace NA by 0 # raster::values to avoid error in knitr, otherwise use:
# chm[is.na(chm[])] <- 0
raster::values(chm)[is.na(raster::values(chm))] <- 0
# replace negative values by 0
chm[chm[] < 0] <- 0
# set maximum threshold at 50m
chm[chm > 50] <- 50
# display CHM and areas where vegetation height is lower than 1m
par(mfrow = c(1, 2))
raster::plot(chm, main = "CHM")
raster::plot(chm < 1, asp = 1, main = "CHM < 1m", legend = FALSE)
```



## Gap detection

Gaps will be detected according to two different sets of criteria:

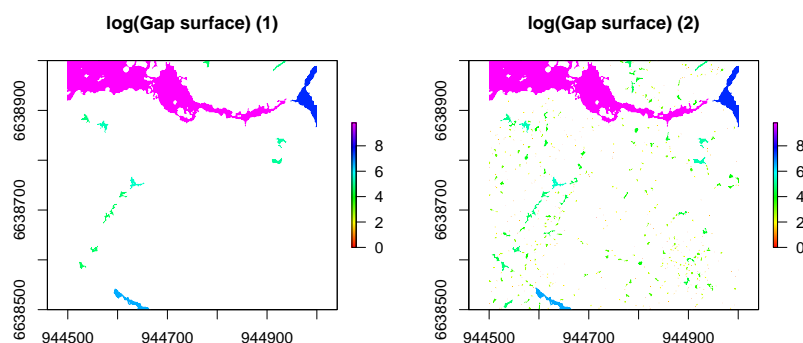
1. Vegetation height lower than 1 m, distance to surrounding vegetation larger than 0.5 times vegetation height and minimum gap surface of 100 m<sup>2</sup>.
2. Vegetation height lower than 1 m, and no restrictions on distance to surrounding vegetation and surface.

The procedure of gap detection is exemplified in Glad et al. (2020).

```
# Perform gap detection with gap width criterion
gaps1 <- lidaRtree::gap_detection(chm, ratio = 2, gap_max_height = 1, min_gap_surface = 100)
# Perform gap detection without gap width nor min gap surface criteria
gaps2 <- lidaRtree::gap_detection(chm, ratio = NULL, gap_max_height = 1, min_gap_surface = 0)
```

The following figure displays the obtained gaps, colored by size.

```
# display CHM and areas where vegetation height is lower than 1m
par(mfrow = c(1, 2))
# max value of surface (log)
z_lim <- log(max(raster::values(gaps1$gap_surface)))
# plot gap surface
raster::plot(log(gaps1$gap_surface), main = "log(Gap surface) (1)", zlim = c(0, z_lim),
  col = rainbow(255, end = 5/6))
raster::plot(log(gaps2$gap_surface), main = "log(Gap surface) (2)", zlim = c(0, z_lim),
  col = rainbow(255, end = 5/6))
```

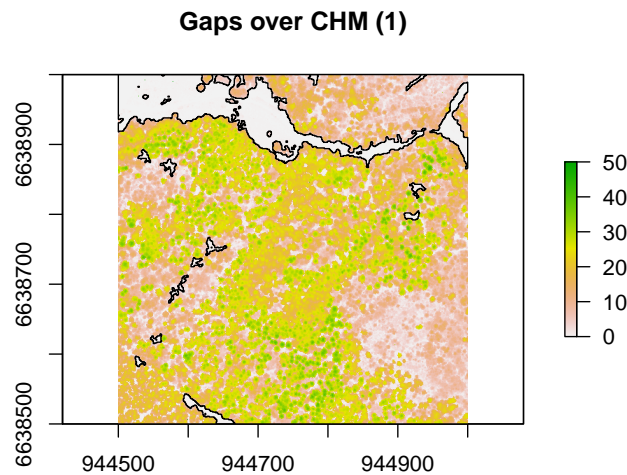


Gaps can be vectorized for display over the original CHM.

```
# convert to vector
gaps1_v <- raster::rasterToPolygons(gaps1$gap_id, dissolve = TRUE)
```

## Le chargement a nécessité le package : rgeos

```
# display gaps over original CHM
raster::plot(chm, main = "Gaps over CHM (1)")
sp::plot(gaps1_v, add = T)
```

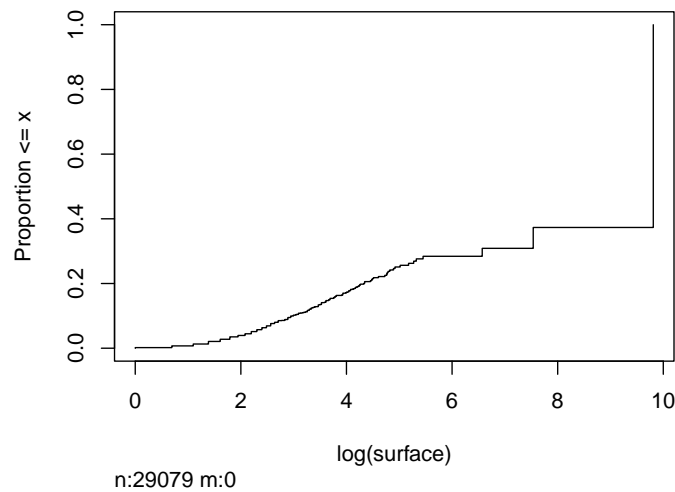


Statistics on gaps (number, size) can be derived from the raster objects. Example for gaps (2)

```
# compute gaps surface
gaps_df <- data.frame(t(table(raster::values(gaps2$gap_id))))[, -1]
# change column names
names(gaps_df) <- c("id", "surface")
# convert surface in pixels to square meters
gaps_df$surface <- gaps_df$surface * raster::res(chm)[1]^2
# data.frame
head(gaps_df, n = 3)
```

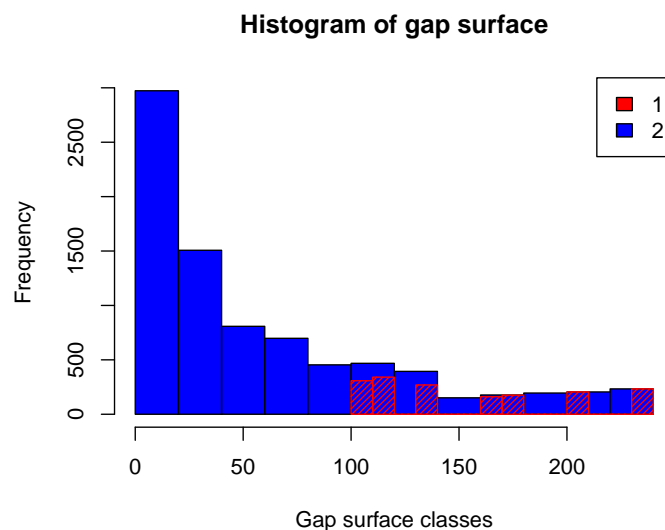
```
##   id surface
## 1  1  18226
## 2  6    26
## 3  7   113
```

```
# cumulative distribution of gap surface by surface
Hmisc::Ecdf(log(gaps_df$surface), weights = gaps_df$surface, xlab = "log(surface)")
```



Histogram can be used to compare the distribution of surface in different classes of gap surface. For better visualization of the differences between the set of criteria, gaps bigger than 500 m<sup>2</sup> are not taken into account.

```
# extract vector of total gap surface of pixel
surface1 <- raster::values(gaps1$gap_surface)
surface2 <- raster::values(gaps2$gap_surface)
# remove large gaps
surface1[surface1 > 500] <- NA
surface2[surface2 > 500] <- NA
# display histogram of surface
hist(surface2, col = "blue", xlab = "Gap surface classes", main = "Histogram of gap surface")
hist(surface1, border = "red", col = "red", density = 30, add = TRUE)
legend("topright", c("1", "2"), fill = c("red", "blue"))
```

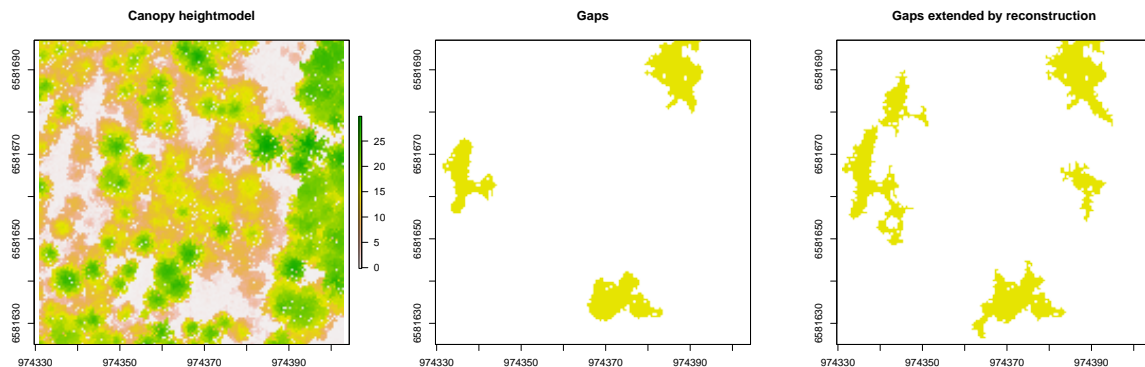


## Edges detection

In the previous part, a pixel that fulfills the maximum height criterion but does not fulfill the distance ratio criterion (i.e. it is too close to surrounding vegetation based on the distance / vegetation height ratio) is removed from gap surface. For edge detection, it seems more appropriate to integrate those

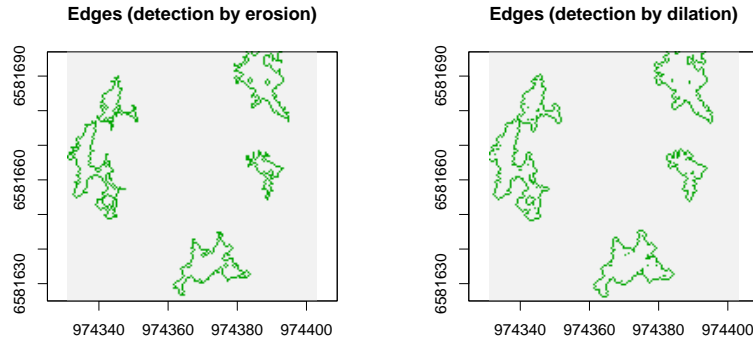
pixels in the gap surface, otherwise some edges would be detected inside flat areas. The difference is exemplified in the following plots. The second image exhibits more gaps, because some gaps which are not reconstructed do not reach the minimum surface. The gaps in the second image are also larger, because gaps extend to all neighboring pixels complying with the height threshold, even if they do not comply with the distance criterion.

```
# load canopy height model
chm <- lidaRtRee::chm_chablais3
chm[is.na(chm)] <- 0
# Perform gap detection with distance ratio criterion
gaps1 <- lidaRtRee::gap_detection(chm, ratio = 2, gap_max_height = 1, min_gap_surface = 50,
  nl_filter = "None")
# Perform gap detection with distance ratio criterion and gap reconstruction
gaps1r <- lidaRtRee::gap_detection(chm, ratio = 2, gap_max_height = 1, min_gap_surface = 50,
  gap_reconstruct = TRUE, nl_filter = "None")
# display detected gaps
par(mfrow = c(1, 3))
# plot gaps
raster::plot(chm, main = "Canopy heightmodel")
raster::plot(gaps1$gap_id > 0, main = "Gaps", legend = FALSE)
raster::plot(gaps1r$gap_id > 0, main = "Gaps extended by reconstruction", legend = FALSE)
```



Edge detection is performed by extraction of the difference between a binary image of gaps and the result of a morphological erosion or dilation applied to the same image.

```
# Perform edge detection by erosion
edge_inside <- lidaRtRee::edge_detection(gaps1r$gap_id > 0)
# Perform edge detection by dilation
edge_outside <- lidaRtRee::edge_detection(gaps1r$gap_id > 0, inside = FALSE)
# display zoom on edges
par(mfrow = c(1, 2))
raster::plot(edge_inside, main = "Edges (detection by erosion)", legend = FALSE)
raster::plot(edge_outside, main = "Edges (detection by dilation)", legend = FALSE)
```



Percentage of edges can be calculated as the ratio between edge pixels surface and total surface, multiplied by 100. The result is dependent on the raster resolution. Obtained percentage for the “erosion” method is 4.1, whereas it is 4 for method “dilation.”

## References

Glad, Anouk, Björn Reineking, Marc Montadert, Alexandra Depraz, and Jean-Matthieu Monnet. 2020. “Assessing the Performance of Object-Oriented LiDAR Predictors for Forest Bird Habitat Suitability Modeling.” *Remote Sensing in Ecology and Conservation* 6 (1): 5–19. <https://doi.org/10.1002/rse2.117>.