R test run of rugosity code

Jeff Atkins

October 5, 2016

```
## Loading required package: ggplot2
```

OVERVIEW

This document runs through the testing of a draft module of an R package to be used to calculate canopy structural metrics from portable canopy LiDAR (PCL) data. This code is meant to reproduce methods as outlined in Hardiman et al. 2011 etc.

The code now pulls from an functions.R script that includes the sub-functions that parse and process the PCL data files.

```
# Source functions
source("functions.R")
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

WORKFLOW

The draft of the script runs a single-transect at a time and requires the entry of a data directory (noted as: `data_dir`) and the name of the file to be imported (noted as: `filename`). The script then uses the function `read.pcl` that uses a custom `read.csv` format to import and structure PCL data. A diagnostic plot of the data can also be produced with the function `pcl.diagnostic.plot`.

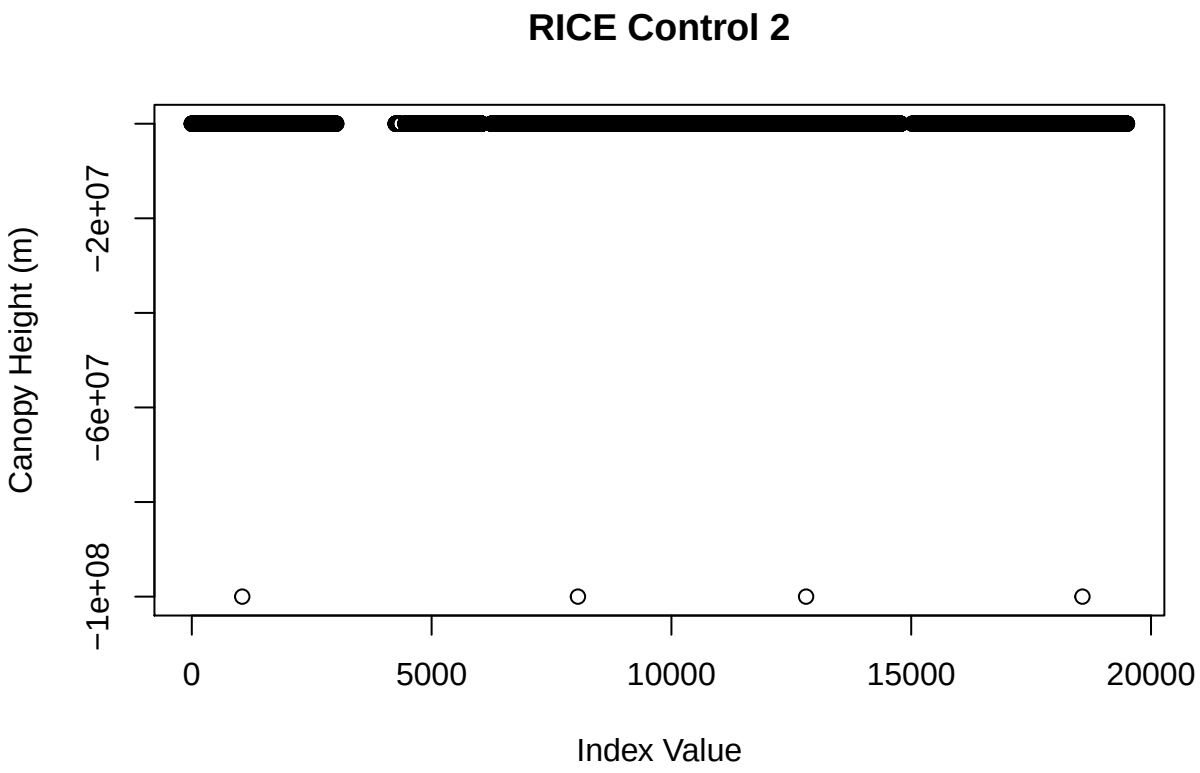
```
data_dir <- "./data/rice/"
filename <- "rice_control_two.CSV"

#function to import data
test.data <- read.pcl(data_dir, filename)

# let's take a quick look at the data
head(test.data)
```

```
##   index return_distance intensity
## 1     1          13.250        76
## 2     2          13.298        75
## 3     3          13.298        75
## 4     4          13.252        78
## 5     5          13.248        73
## 6     6          13.298        75
```

```
pcl.diagnostic.plot(test.data, "RICE Control 2", -1e+08)
```



The next step includes adjusting by user height with the function `adjust_by_user` that takes the data frame name and the "Brady Height" value (the distance of the PCL from the ground in meters). LiDAR "hits" are then coded using `code_hits`–a function that allows for processing the data through vectorization. The data are then binned to x and z bins using the function `split_transects_from_pcl`.

```
test.data <- adjust_by_user(test.data, 1.2)

test.data <- code_hits(test.data)

#the bread and butter of this code
test.data.binned <- split_transects_from_pcl(test.data, 30, 10)

#let's look at the data again
head(test.data.binned)
```

```
##      index return_distance intensity sky_hit can_hit marker seg_num
```

```
## 1053  1053          12.046        63   FALSE    TRUE  FALSE       1
## 1054  1054          12.046        60   FALSE    TRUE  FALSE       1
## 1055  1055          12.046        57   FALSE    TRUE  FALSE       1
## 1056  1056          12.146        58   FALSE    TRUE  FALSE       1
## 1057  1057          12.045        61   FALSE    TRUE  FALSE       1
## 1058  1058          11.947        65   FALSE    TRUE  FALSE       1
##      chunk_num xbin zbin
## 1053         1    1   13
## 1054         1    1   13
## 1055         1    1   13
## 1056         1    1   13
## 1057         1    1   13
## 1058         1    1   12
```

```
summary(test.data.binned)
```

```
##      index       return_distance       intensity       sky_hit
##  Min.   : 1053   Min.   :-99999998   Min.   :  8.00   Mode :logical
##  1st Qu.: 5432   1st Qu.:        2   1st Qu.: 53.00   FALSE:15629
##  Median : 9812   Median :        3   Median : 59.00   TRUE :1889
##  Mean   : 9812   Mean   :   -19190   Mean   : 59.74   NA's :0
##  3rd Qu.:14191   3rd Qu.:        6   3rd Qu.: 68.00
##  Max.   :18570   Max.   :       25   Max.   :106.00
##                  NA's   :1889        NA's   :1892
##   can_hit          marker           seg_num        chunk_num
##  Mode :logical   Mode :logical   Min.   :1.000   Min.   : 1.0
##  FALSE:1892      FALSE:17515     1st Qu.:1.000   1st Qu.: 3.0
##  TRUE :15626     TRUE :3         Median :2.000   Median : 5.0
##  NA's :0         NA's :0         Mean   :1.929   Mean   : 5.5
##                                  3rd Qu.:3.000   3rd Qu.: 8.0
##                                  Max.   :3.000   Max.   :10.0
##
##       xbin            zbin
##  Min.   : 1.00   Min.   :-99999997
##  1st Qu.: 7.00   1st Qu.:        3
##  Median :14.00   Median :        4
##  Mean   :14.79   Mean   :   -17121
##  3rd Qu.:23.00   3rd Qu.:        6
##  Max.   :30.00   Max.   :       26
##
```

Next comes the formulation of the matrix for the basis of further calculations. The matrix results from condensing data into a matrix with dimensions of length of transect X maximum measured canopy height. For example, a 40 m transect with max canopy height of 29.3 m would results in a matrix with dimensions of 40 x 29. The function `make_matrix` that does this gives some extra info as well to be cleaned up.

```
m <- make_matrix(test.data.binned)
summary(m)
```

```
##       xbin           zbin     lidar.pulses       bin.hits
##  Min.   : 1.0   Min.   : 0   Min.   :  0.00   Min.   :  0.00
##  1st Qu.: 8.0   1st Qu.: 6   1st Qu.:  0.00   1st Qu.:  0.00
```

```
##  Median :15.5   Median :13   Median :  0.00   Median :  0.00
##  Mean   :15.5   Mean   :13   Mean   : 89.52   Mean   : 19.29
##  3rd Qu.:23.0   3rd Qu.:20   3rd Qu.:  0.00   3rd Qu.:  0.00
##  Max.   :30.0   Max.   :26   Max.   :700.00   Max.   :699.00
##     sky.hits          can.hits         mean.ht           sd.ht
##  Min.   :  0.000   Min.   :  0.00   Min.   : 0.0000   Min.   :0.0000
##  1st Qu.:  0.000   1st Qu.:  0.00   1st Qu.: 0.0000   1st Qu.:0.0000
##  Median :  0.000   Median :  0.00   Median : 0.0000   Median :0.0000
##  Mean   :  5.947   Mean   : 83.56   Mean   : 0.8713   Mean   :0.4004
##  3rd Qu.:  0.000   3rd Qu.:  0.00   3rd Qu.: 0.0000   3rd Qu.:0.0000
##  Max.   :700.000   Max.   :700.00   Max.   :16.3571   Max.   :9.2047
##      max.ht            Freq
##  Min.   : 0.000   Min.   :1
##  1st Qu.: 0.000   1st Qu.:1
##  Median : 0.000   Median :1
##  Mean   : 2.064   Mean   :1
##  3rd Qu.: 0.000   3rd Qu.:1
##  Max.   :25.100   Max.   :1
```

Now we calculate VAI using the following function `calc_vai` which is included for demo purposes:

```
calc_vai <- function(df) {

     df$vai <- (df$bin.hits / df$lidar.pulses)
     df$vai <- df$vai * 8 #adjust for max lai?
     df$vai <- df$vai * -1
     df$vai <- log(1.0 - df$vai*0.9817)/0.5
     df[is.na(df)] <- 0
     return(df)

}
```

```
m <- calc_vai(m)
```

Then comes Rugosity, but it is the end of the day and I need to get home so here is the rest which also includes a draft hit grid:

```
#######new test
m <- calc_mean_leaf_ht(m)
m <- calc_std_bin(m)

#the jam
calc_rugosity(m)
```

```
## Square of leaf height variance (stdStd from old script)
```

```
## [1] 537.0238
```

```
## Mean Standard deviation of leaf heights
```

```
## [1] 9.516462
```

```
## Maximum VAI
```

```
## [1] 4.359112
```

```
## Canopy Rugosity
```

```
## [1] 21.12962
```

```
#hit grid
ggplot(m, aes(x = xbin, y = zbin))+
     geom_tile(aes(fill = vai))+
     scale_fill_gradient(low="palegreen1", high="dark green",
                         name="LiDAR\n Method One")+
     #scale_y_continuous(breaks = seq(0, 20, 5))+
     # scale_x_continuous(minor_breaks = seq(0, 40, 1))+
     theme(axis.line = element_line(colour = "black"),
           panel.grid.major = element_blank(),
           panel.grid.minor = element_blank(),
           panel.background = element_blank())+
     xlim(0,40)+
     ylim(0,40)+
     xlab("Distance along transect (m)")+
     ylab("Height above ground (m)")
```