Site Specific LiDAR Data

Conor McMahon

2/23/2022

We're going to extract LiDAR statistics around survey points for both 2015 and 2018.

First, we'll define some functions to get the height histogram in a local region around a point, and also a histogram of the CHM around each point.

```
{

  getNeighborhood <- function(input_image, point, buffer_radius)
  {
    # reproject POINT to raster CRS
    point <- spTransform(point, crs(input_image))
    # extract data in BUFFER_RADIUS distance around POINT
    masked_data <- raster::extract(input_image, point, buffer=buffer_radius)
  }
}
```

Load LiDAR data - both CHM rasters (including maximum height in each 1 m pixel) and histograms with all return values (including number of returns at various heights within each 10 m pixel).

```
# Load LiDAR raster data
histogram_2015 <- stack(here::here("outputs","histogram_2015_mosaic.tif"))
histogram_2018 <- stack(here::here("outputs","histogram_2018_mosaic.tif"))
chm_2015 <- raster(here::here("outputs","chm_mosaic_2015_filtered.tif"))
chm_2018 <- raster(here::here("outputs","chm_mosaic_2018_filtered.tif"))
```

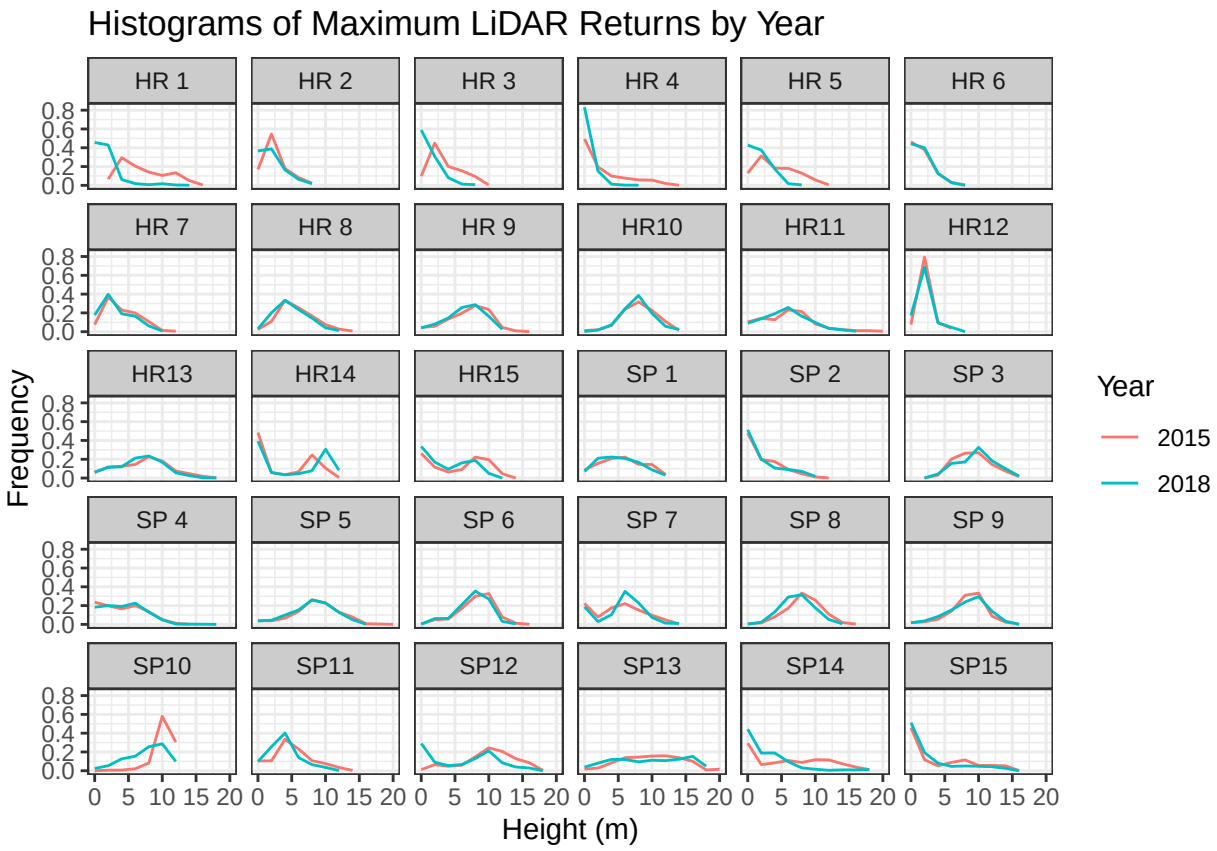
Analyze survey points, using 50 m radii around each point. First we'll look only at maximum returns within each 1m pixel window:

```
# Load and apply initial CRS to survey points (Lat/Lon values)
survey_points <- read_csv(here::here("birds","Point_count_station_list.csv"))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Project = col_character(),
##   Property = col_character(),
##   `Location/Type` = col_character(),
##   `Station #` = col_character(),
##   Latitude = col_double(),
##   Longitude = col_double()
## )
```

```r
coordinates(survey_points) <- ~Longitude+Latitude
crs(survey_points) <- crs("+init=epsg:4326")

# Get all CHM values within 50 m radius of each survey point
#   In 2015:
survey_max_returns_2015 <- getNeighborhood(chm_2015, survey_points, buffer_radius = 50)
#   In 2018:
survey_max_returns_2018 <- getNeighborhood(chm_2018, survey_points, buffer_radius = 50)

# Function to bin to a histogram of maximum return height within each 1 m pixel in 50 m radius around a
binLiDARReturns <- function(height_vector, min_height, max_height, bins, year)
{
  # Upper end of each bin
  bin_breaks <- (1:bins)*(max_height-min_height)/bins
  # Lower end of each bin
  bin_starts <- (1:bins-1)*(max_height-min_height)/bins
  # Generate histogram data frame
  histogram_df <- data.frame(height = height_vector,
                             bin_index = .bincode(height_vector, bin_starts)) %>%
    mutate(bin_start = bin_starts[bin_index],
           bin_end = bin_breaks[bin_index]) %>%
    group_by(bin_index) %>%
    summarize(count = n(),
              bin_index = median(bin_index),
              bin_start = median(bin_start),
              bin_end = median(bin_end)) %>%
    mutate(year = rep(year, n()))
  # Add normalized frequency
  histogram_df$frequency <- histogram_df$count / sum(histogram_df$count)
  return(histogram_df)
}

# Apply CHM (max height) binning function for each survey point
#   In 2015:
chm_histograms_2015 <- lapply(survey_max_returns_2015, binLiDARReturns, min_height=0, max_height=30, bi
#   In 2018:
chm_histograms_2018 <- lapply(survey_max_returns_2018, binLiDARReturns, min_height=0, max_height=30, bi
# The above are lists of data frames... add survey point labels to each data frame:
for(ind in 1:length(chm_histograms_2015))
{
  chm_histograms_2015[[ind]]$site <- rep(survey_points[ind,]$`Station #`, nrow(chm_histograms_2015[[ind]
  chm_histograms_2018[[ind]]$site <- rep(survey_points[ind,]$`Station #`, nrow(chm_histograms_2018[[ind]
}
# Combine list of dataframes to one dataframe
chm_histograms_all <- rbind(bind_rows(chm_histograms_2015), bind_rows(chm_histograms_2018))

# Generate plot of CHM histograms by site and year
ggplot(chm_histograms_all) +
    theme_bw() +
  theme(strip.background = element_rect(fill = 'gray80')) +
    geom_line(aes(x=bin_start, y=frequency, group=year, col=as.factor(year))) +
    facet_wrap(~site) +
    scale_x_continuous(limits=c(0,20)) +
```

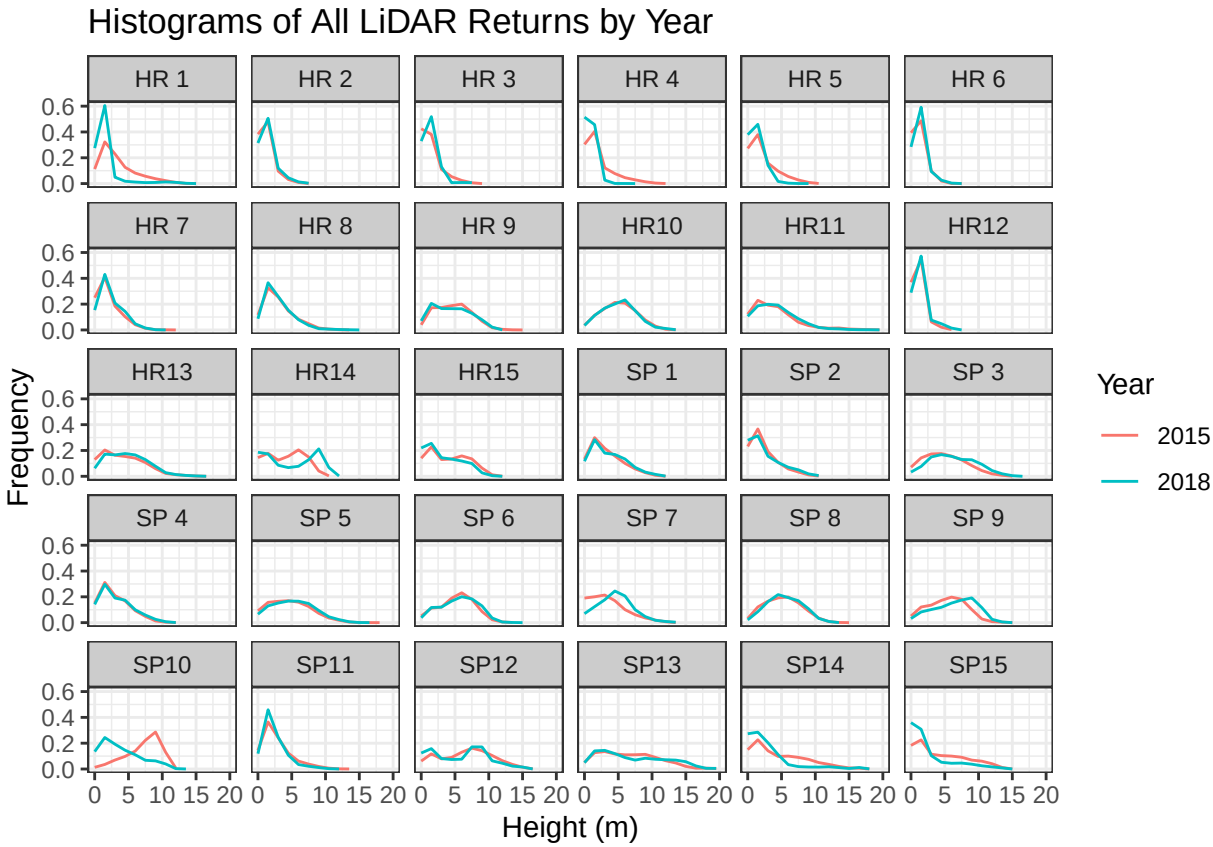
```
    ggtitle("Histograms of Maximum LiDAR Returns by Year") + xlab("Height (m)") +
    ylab("Frequency") +
    guides(col=guide_legend(title="Year"))
```



Now we want to repeat the above, but considering ALL returns, not just the highest ones:

```
# Get matrices where rows rows are pixels in the histogram image, columns are height bins, and cells ar
#    In 2015:
hist_neighborhood_2015 <- getNeighborhood(histogram_2015, survey_points, buffer_radius = 50)
#    In 2018
hist_neighborhood_2018 <- getNeighborhood(histogram_2018, survey_points, buffer_radius = 50)
# Convert the above to a list of height frequency averages across all neighbor pixels in image:
collapseNeighborhood <- function(neighborhood)
{
  # Drop first two columns (which include min/max values in histogram), sum across pixels (columns)
  num_bins <- ncol(neighborhood) - 2
  frequency_sums <- as.numeric(colSums(neighborhood))[3:(num_bins+2)]
  normalized_frequency <- frequency_sums / (sum(frequency_sums))
  # Set up bin to height conversion
  min_value <- median(neighborhood[,1])
  max_value <- median(neighborhood[,2])
  bin_breaks = (1:num_bins)*(max_value-min_value)/num_bins + min_value
  bin_starts = (0:(num_bins-1))*(max_value-min_value)/num_bins + min_value
  bin_indices = 1:num_bins
  return ( data.frame(bin_start = bin_starts,
                      bin_break = bin_breaks,
```

```
                      bin_index = bin_indices,
                      frequency = normalized_frequency) )
}
return_frequencies_2015 <- lapply(hist_neighborhood_2015, collapseNeighborhood)
return_frequencies_2018 <- lapply(hist_neighborhood_2018, collapseNeighborhood)
# Add site information
for(ind in 1:length(return_frequencies_2015))
{
  return_frequencies_2015[[ind]]$site <- rep(survey_points[ind,]$`Station #`, nrow(return_frequencies_2
  return_frequencies_2015[[ind]]$year <- rep(2015, nrow(return_frequencies_2015[[ind]]))
  return_frequencies_2018[[ind]]$site <- rep(survey_points[ind,]$`Station #`, nrow(return_frequencies_2
  return_frequencies_2018[[ind]]$year <- rep(2018, nrow(return_frequencies_2015[[ind]]))
}
# Combine list of dataframes to one dataframe
total_histograms_all <- rbind(bind_rows(return_frequencies_2015), bind_rows(return_frequencies_2018))

# Replace 0 frequencies with NA to clarify graphs
total_histograms_all[total_histograms_all$frequency == 0,]$frequency <- NA

# Generate plot of all return values by site and year
ggplot(total_histograms_all) +
    theme_bw() +
  theme(strip.background = element_rect(fill = 'gray80')) +
    geom_line(aes(x=bin_start, y=frequency, group=year, col=as.factor(year))) +
    facet_wrap(~site) +
    scale_x_continuous(limits=c(0,20)) +
    ggtitle("Histograms of All LiDAR Returns by Year") + xlab("Height (m)") +
    ylab("Frequency") +
    guides(col=guide_legend(title="Year"))
```



Now lets do some comparisons to bird data:

```
birds <- rbind(read_csv(here::here("birds","SP_abundance_breeding_2015_2021.csv")) ) %>%#,
   #            read_csv(here::here("birds","SP_abundance_nonbreeding_2015_2021.csv"))) %>%
  janitor::clean_names()
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Year = col_double(),
##   Site_code = col_character(),
##   Species = col_character(),
##   TotalBirds = col_double(),
##   `Relative Abund Index` = col_double(),
##   Season = col_character()
## )
```

```
# Bird sites use a different naming convention than survey points - correct here (e.g. PC-01 is SP 1)
site_name_remapping <- c(paste("SP ",1:9,sep=""), paste("SP",10:15,sep=""))
names(site_name_remapping) <- c(paste("PC-0",1:9,sep=""),paste("PC-",10:15,sep=""))
birds$site <- as.character(site_name_remapping[birds$site_code])

# Reformat histograms to be wide:
chm_histograms_wide <- chm_histograms_all %>%
  dplyr::select(bin_index,frequency,year,site) %>%          # drop extra bin information
```

```
  drop_na() %>%
  pivot_wider(names_from=bin_index, values_from=frequency)  # switch to wide format - bins as columns
# Re-order columns so bins are in sensible order
chm_histograms_wide <- chm_histograms_wide %>%
  dplyr::select(c(1,2,2+(order(as.numeric(colnames(chm_histograms_wide)[3:ncol(chm_histograms_wide)])))
  replace(is.na(.), 0)

total_histograms_wide <- total_histograms_all %>%
  dplyr::select(bin_index,frequency,year,site) %>%          # drop extra bin information
  drop_na() %>%
  pivot_wider(names_from=bin_index, values_from=frequency)  # switch to wide format - bins as columns
# Re-order columns so bins are in sensible order
total_histograms_wide <- total_histograms_wide %>%
  dplyr::select(c(1,2,2+(order(as.numeric(colnames(chm_histograms_wide)[3:ncol(total_histograms_wide)])))
  replace(is.na(.), 0)


# Lets pick some target birds based on how many records there are for the species (can't represent spec
bird_data_summary <- birds %>%
  group_by(species) %>%
  summarize(total_count = sum(total_birds),
            avg_abund_index = mean(relative_abund_index),
            presence_count = n())
view(bird_data_summary)

yewa_data <- birds %>%
  filter(year %in% c(2015, 2018),
         species == "LBVI")

bird_lidar_data <- merge(chm_histograms_wide, yewa_data, by=c("site","year"), all.x=TRUE) %>%
  filter(substr(site,1,1)=="S") %>%
  replace(is.na(.), 0)
names(bird_lidar_data)  <- c("site","year",
                  paste("bin_",names(bird_lidar_data[3:16]),sep=""),
                  "site_code", "species", "total_birds", "relative_abund_index")
bird_lidar_data$quad_1 <- bird_lidar_data$bin_1
bird_lidar_data$quad_2 <- bird_lidar_data$bin_2 + bird_lidar_data$bin_3 + bird_lidar_data$bin_4 + bird_l
bird_lidar_data$quad_3 <- bird_lidar_data$bin_6 + bird_lidar_data$bin_7 + bird_lidar_data$bin_8 + bird_l
bird_lidar_data$quad_4 <- bird_lidar_data$bin_10 + bird_lidar_data$bin_11 + bird_lidar_data$bin_12
bird_model <- lm(data=bird_lidar_data, relative_abund_index~quad_1+quad_2+quad_3+quad_4-1)
summary(bird_model)
```

```
##
## Call:
## lm(formula = relative_abund_index ~ quad_1 + quad_2 + quad_3 +
##     quad_4 - 1, data = bird_lidar_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.58206 -0.31170 -0.08586  0.24355  1.01550
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
```

```
## quad_1  -0.2858     0.4654  -0.614  0.54440
## quad_2   0.7221     0.2381   3.033  0.00544 **
## quad_3   0.4223     0.3227   1.309  0.20215
## quad_4   0.8854     8.3326   0.106  0.91619
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4554 on 26 degrees of freedom
## Multiple R-squared:  0.5847, Adjusted R-squared:  0.5209
## F-statistic: 9.153 on 4 and 26 DF,  p-value: 9.39e-05
```

```
plot(predict(bird_model,bird_lidar_data), bird_lidar_data$total_birds)
```