

Extract raster values using vector boundaries in R

Learning Objectives

After completing this tutorial, you will be able to:

- Use the `extract()` function to extract raster values using a vector extent or set of extents.
- Create a scatter plot with a one to one line in R.
- Understand the concept of uncertainty as it's associated with remote sensing data.

What you need

You will need a computer with internet access to complete this lesson and the data for week 5 of the course.

Download Week 5 Data (~500 MB){:data-proofer-ignore=" .btn }

```
# load libraries
library(raster)
library(rgdal)
library(ggplot2)
library(dplyr)

options(stringsAsFactors = FALSE)

# set working directory
# setwd("path-here/earth-analytics")
```

Import Canopy Height Model

First, we will import a canopy height model created by the NEON project. In the previous lessons / weeks we learned how to make a canopy height model by subtracting the Digital elevation model (DEM) from the Digital surface model (DSM).

```
# import canopy height model (CHM).
SJER_chm <- raster("data/week5/california/SJER/2013/lidar/SJER_lidarCHM.tif")
SJER_chm
## class      : RasterLayer
## dimensions  : 5059, 4296, 21733464  (nrow, ncol, ncell)
## resolution  : 1, 1  (x, y)
## extent     : 254571, 258867, 4107303, 4112362  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=utm +zone=11 +datum=WGS84 +units=m +no_defs +ellps=WGS84 +towgs84=0,0,0
## data source : /Users/lewa8222/Documents/earth-analytics/data/week5/california/SJER/2013/lidar/SJER_lidarCHM.tif
## names      : SJER_lidarCHM
## values     : 0, 45.88  (min, max)

# plot the data
hist(SJER_chm,
     main="Histogram of Canopy Height\n NEON SJER Field Site",
     col="springgreen",
     xlab="Height (m)")
## Warning in .hist1(x, maxpixels = maxpixels, main = main, plot = plot, ...):
## 0% of the raster cells were used. 100000 values used.
```

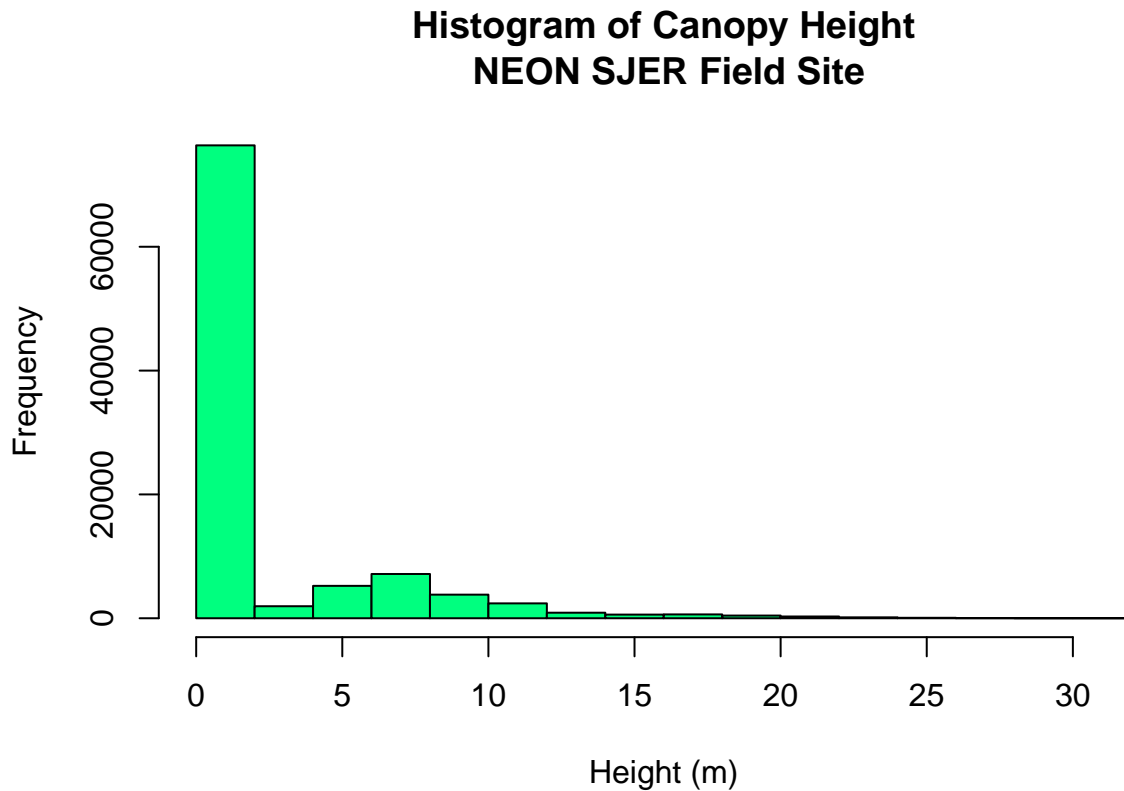


Figure 1: Histogram of CHM values

There are a lot of values in our CHM that == 0. Let's set those to NA and plot again.

```
# set values of 0 to NA as these are not trees
SJER_chm[SJER_chm==0] <- NA

# plot the modified data
hist(SJER_chm,
     main="Histogram of Canopy Height\n pixels==0 set to NA",
     col="springgreen",
     xlab="Height (m)")
```

Part 2. Does our CHM data compare to field measured tree heights?

We now have a canopy height model for our study area in California. However, how do the height values extracted from the CHM compare to our laboriously collected, field measured canopy height data? To figure this out, we will use *in situ* collected tree height data, measured within circular plots across our study area. We will compare the maximum measured tree height value to the maximum LiDAR derived height value for each circular plot using regression.

For this activity, we will use the a csv (comma separate value) file, located in SJER/2013/insitu/veg_structure/D17_2013_SJ

```
# import plot centroids
SJER_plots <- readOGR("data/week5/california/SJER/vector_data",
                    "SJER_plot_centroids")

## OGR data source with driver: ESRI Shapefile
## Source: "data/week5/california/SJER/vector_data", layer: "SJER_plot_centroids"
```

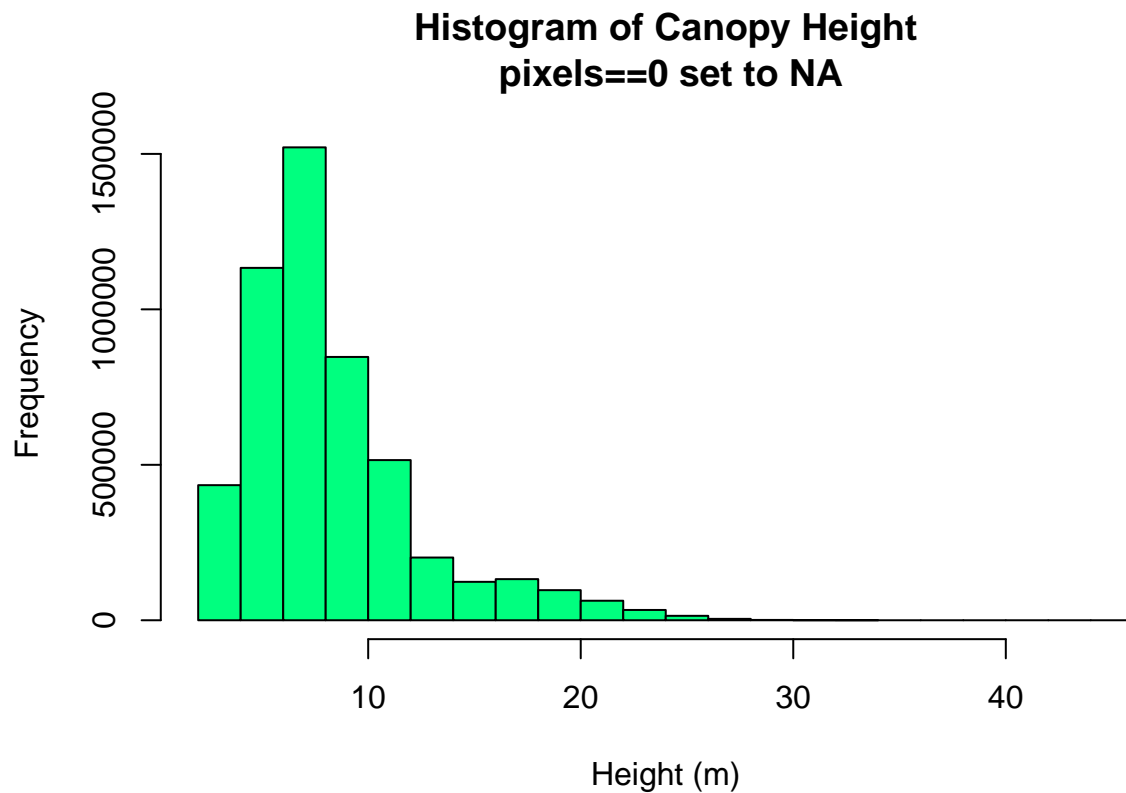


Figure 2: histogram of chm values

```
## with 18 features
## It has 5 fields

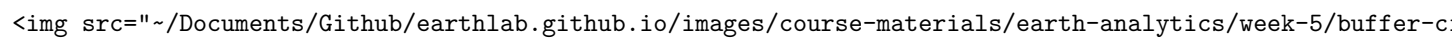
# Overlay the centroid points and the stem locations on the CHM plot
plot(SJER_chm,
     main="SJER Plot Locations",
     col=gray.colors(100, start=.3, end=.9))

# pch 0 = square
plot(SJER_plots,
     pch = 16,
     cex = 2,
     col = 2,
     add=TRUE)
```

Extract CMH data within 20 m radius of each plot centroid.

Next, we will create a boundary region (called a buffer) representing the spatial extent of each plot (where trees were measured). We will then extract all CHM pixels that fall within the plot boundary to use to estimate tree height for that plot.

There are a few ways to go about this task. If our plots are circular, then we can use the `extract()` function.

The `extract` function in R allows you to specify a circular buffer

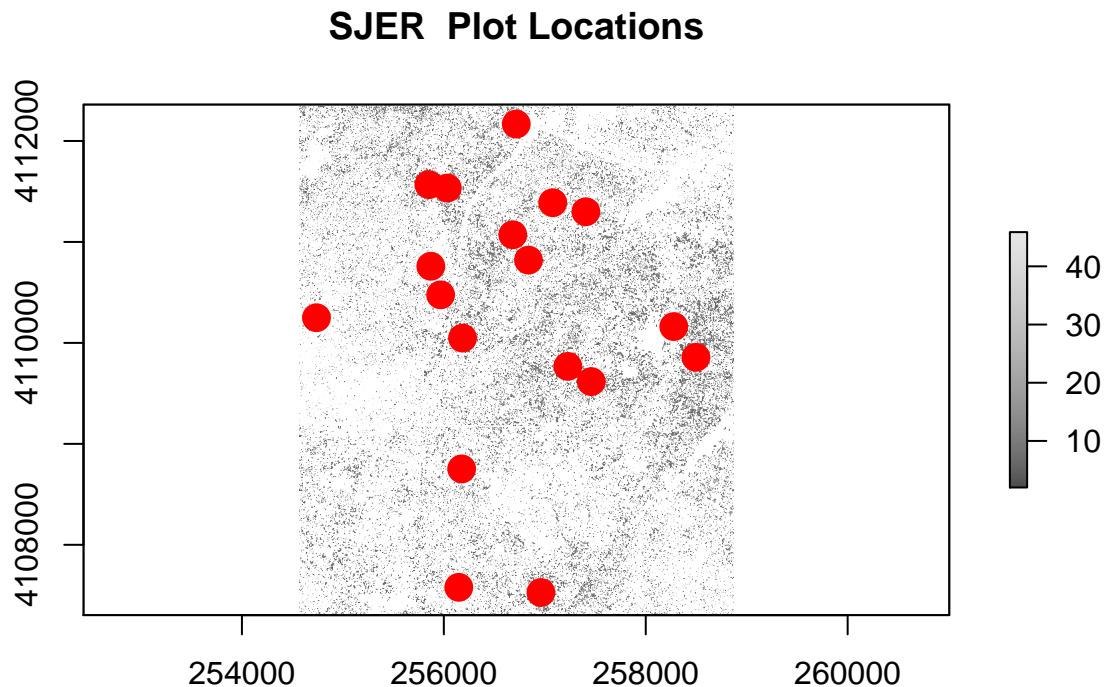


Figure 3: canopy height model / plot locations plot

radius around an x,y point location. Values for all pixels in the specified raster that fall within the circular buffer are extracted. In this case, we will tell R to extract the maximum value of all pixels using the `fun=max` command. Source: Colin Williams, NEON

</figcaption>

Extract Plot Data Using Circle: 20m Radius Plots

```
# Insitu sampling took place within 40m x 40m square plots, so we use a 20m radius.
# Note that below will return a data.frame containing the max height
# calculated from all pixels in the buffer for each plot
SJER_height <- extract(SJER_chm,
  SJER_plots,
  buffer = 20, # specify a 20 m radius
  fun=mean, # extract the MEAN value from each plot
  sp=TRUE, # create spatial object
  stringsAsFactors=FALSE)
```

Explore The Data Distribution

If you want to explore the data distribution of pixel height values in each plot, you could remove the `fun` call to `max` and generate a list. `cent_ovrList <- extract(chm,centroid_sp,buffer = 20)`. It's good to look at the distribution of values we've extracted for each plot. Then you could generate a histogram for each plot `hist(cent_ovrList[[2]])`. If we wanted, we could loop through several plots and create histograms using a `for` loop.

```
# cent_ovrList <- extract(chm,centroid_sp,buffer = 20)
# create histograms for the first 5 plots of data
```

```
# for (i in 1:5) {
#   hist(cent_ourList[[i]], main=paste("plot",i))
# }
```

Derive Square Plot boundaries, then CHM values around a point

For how to extract square plots using a plot centroid value, check out the extracting square shapes activity .

%
 group_by(plotid) %>%
 summarise(insitu_max = max(stemheight), insitu_avg = mean(stemheight))

view the data frame to make sure we're happy with the column names.
head(insitu_stem_height)
A tibble: 6 × 3
plotid insitu_max insitu_avg
<chr> <dbl> <dbl>
1 SJER1068 19.3 3.866667
2 SJER112 23.9 8.221429
3 SJER116 16.0 8.218750
```

```
4 SJER117 11.0 6.512500
5 SJER120 8.8 7.600000
6 SJER128 18.2 5.211765
```

## Merge InSitu Data With Spatial data.frame

Once we have our summarized insitu data, we can `merge` it into the centroids `data.frame`. Merge requires two `data.frames` and the names of the columns containing the unique ID that we will merge the data on. In this case, we will merge the data on the `plot_id` column. Notice that it's spelled slightly differently in both `data.frames` so we'll need to tell R what it's called in each `data.frame`.

```
merge the insitu data into the centroids data.frame
```

```
SJER_height <- merge(SJER_height,
 insitu_stem_height,
 by.x = 'Plot_ID',
 by.y = 'plotid')
```

```
SJER_height@data
```

```
Plot_ID Point northing easting plot_type SJER_lidarCHM insitu_max
1 SJER1068 center 4111568 255852.4 trees 11.544348 19.3
2 SJER112 center 4111299 257407.0 trees 10.355685 23.9
3 SJER116 center 4110820 256838.8 grass 7.511956 16.0
4 SJER117 center 4108752 256176.9 trees 7.675347 11.0
5 SJER120 center 4110476 255968.4 grass 4.591176 8.8
6 SJER128 center 4111389 257078.9 trees 8.979005 18.2
7 SJER192 center 4111071 256683.4 grass 7.240118 13.7
8 SJER272 center 4112168 256717.5 trees 7.103862 12.4
9 SJER2796 center 4111534 256034.4 soil 6.405240 9.4
10 SJER3239 center 4109857 258497.1 soil 6.009128 17.9
11 SJER36 center 4110162 258277.8 trees 6.516288 9.2
12 SJER361 center 4107527 256961.8 grass 13.899027 11.8
13 SJER37 center 4107579 256148.2 trees 7.109851 11.5
14 SJER4 center 4109767 257228.3 trees 5.032620 10.8
15 SJER8 center 4110249 254738.6 trees 3.024286 5.2
16 SJER824 center 4110048 256185.6 soil 7.738203 26.5
17 SJER916 center 4109617 257460.5 soil 11.181955 18.4
18 SJER952 center 4110759 255871.2 grass 4.149286 7.7
insitu_avg
1 3.866667
2 8.221429
3 8.218750
4 6.512500
5 7.600000
6 5.211765
7 6.769565
8 6.819048
9 5.085714
10 3.920833
11 9.200000
12 2.451429
13 7.350000
14 5.910526
15 1.057143
```

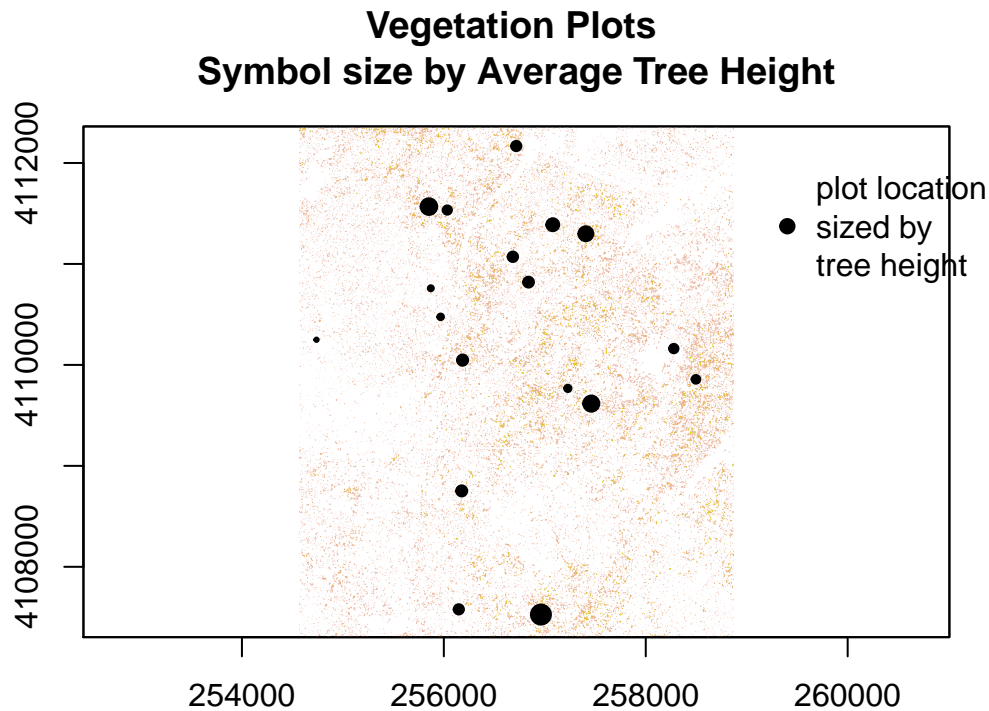


Figure 4: Plots sized by vegetation height

```
16 5.357895
17 5.791667
18 1.558333
```

### Plot by height

```
plot canopy height model
plot(SJER_chm,
 main="Vegetation Plots \nSymbol size by Average Tree Height",
 legend=F)

add plot location sized by tree height
plot(SJER_height,
 pch=19,
 cex=(SJER_height$SJER_lidarCHM)/10, # size symbols according to tree height attribute normalized by
 add=T)

place legend outside of the plot
par(xpd=T)
legend(SJER_chm@extent@xmax+250,
 SJER_chm@extent@ymax,
 legend="plot location \nsized by \ntree height",
 pch=19,
 bty='n')
```

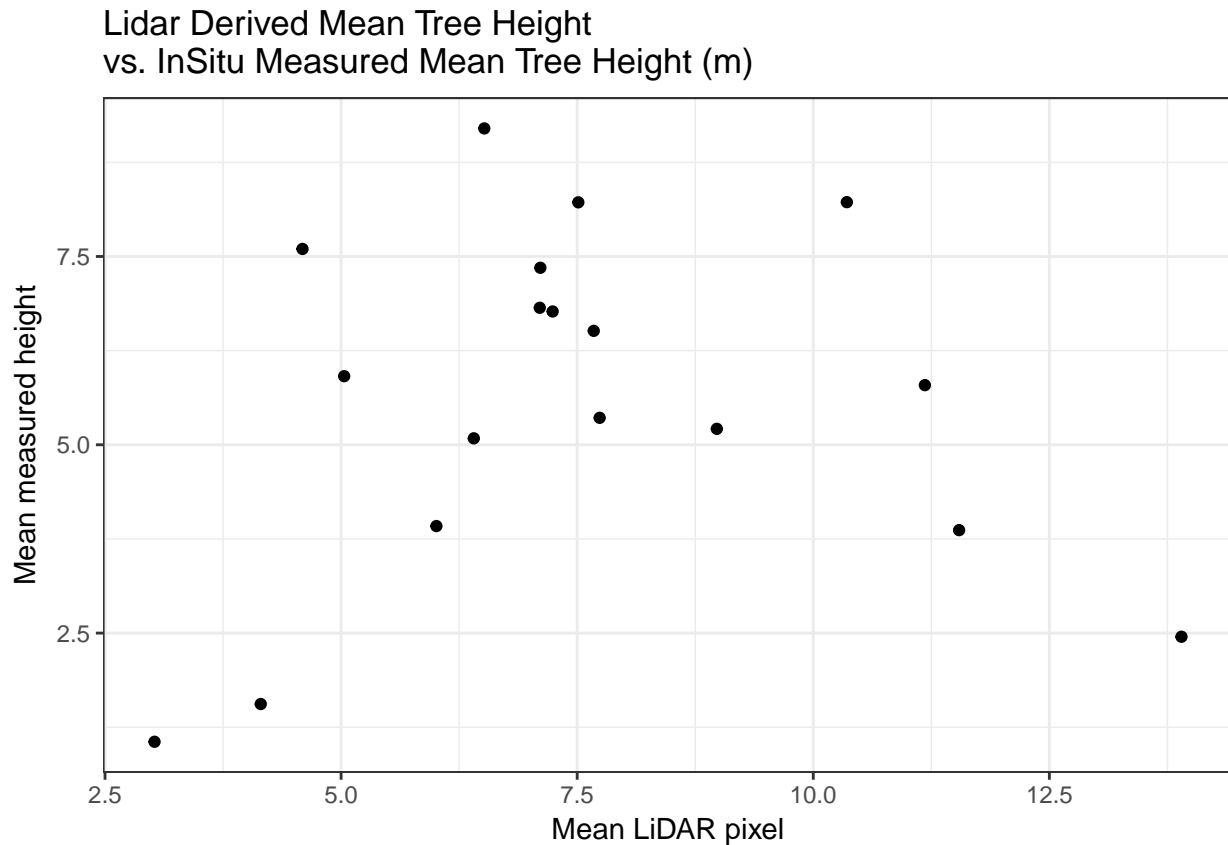


Figure 5: ggplot - measured vs lidar chm.

### Plot Data (CHM vs Measured)

Let's create a plot that illustrates the relationship between in situ measured max canopy height values and lidar derived max canopy height values.

```
create plot
ggplot(SJER_height@data, aes(x=SJER_lidarCHM, y = insitu_avg)) +
 geom_point() +
 theme_bw() +
 ylab("Mean measured height") +
 xlab("Mean LiDAR pixel") +
 ggtitle("Lidar Derived Mean Tree Height \nvs. InSitu Measured Mean Tree Height (m)")
```

Next, let's fix the plot adding a 1:1 line and making the x and y axis the same .

```
create plot
ggplot(SJER_height@data, aes(x=SJER_lidarCHM, y = insitu_avg)) +
 geom_point() +
 theme_bw() +
 ylab("Mean measured height") +
 xlab("Mean LiDAR pixel") +
 xlim(0,15) + ylim(0,15) + # set x and y limits to 0-20
 geom_abline(intercept = 0, slope=1) + # add one to one line
 ggtitle("Lidar Derived Tree Height \nvs. InSitu Measured Tree Height")
```



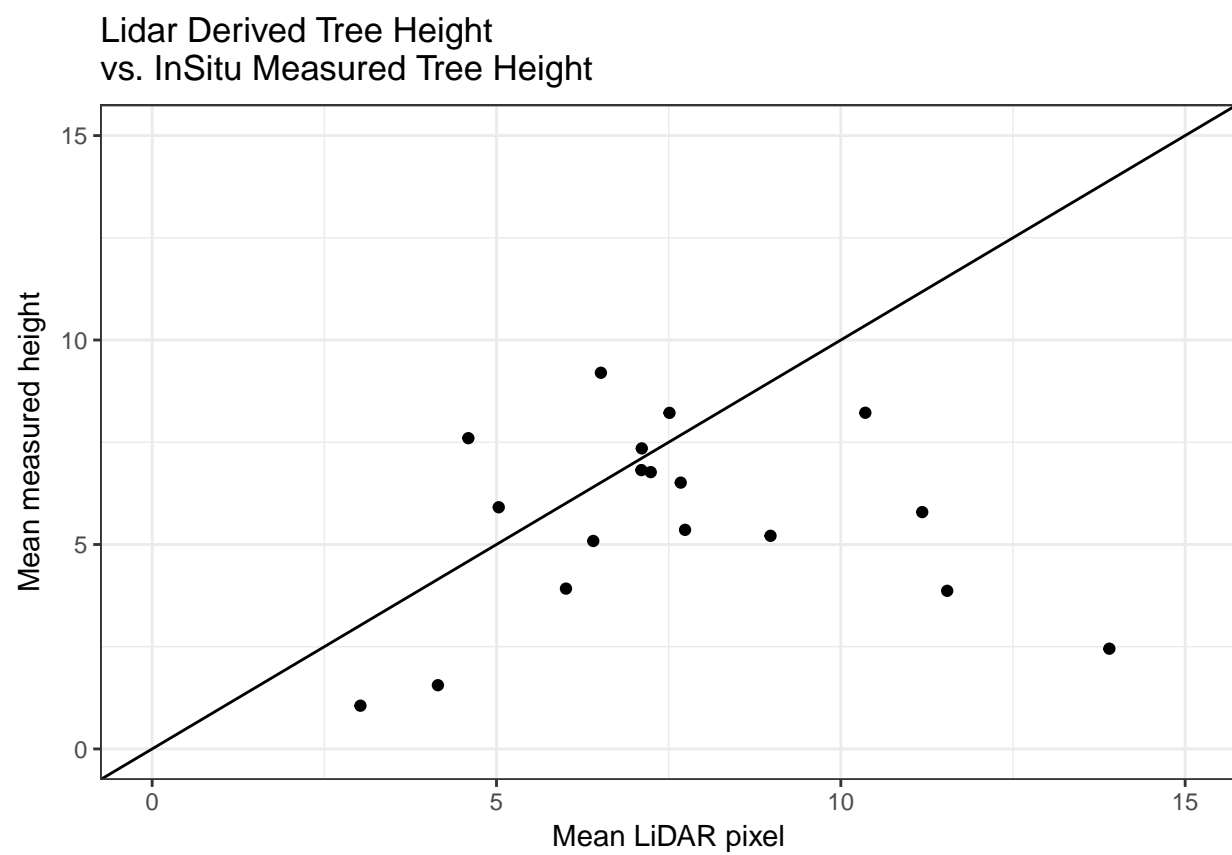


Figure 6: ggplot - measured vs lidar chm w one to one line.

# LiDAR CHM Derived vs Measured Tree Height

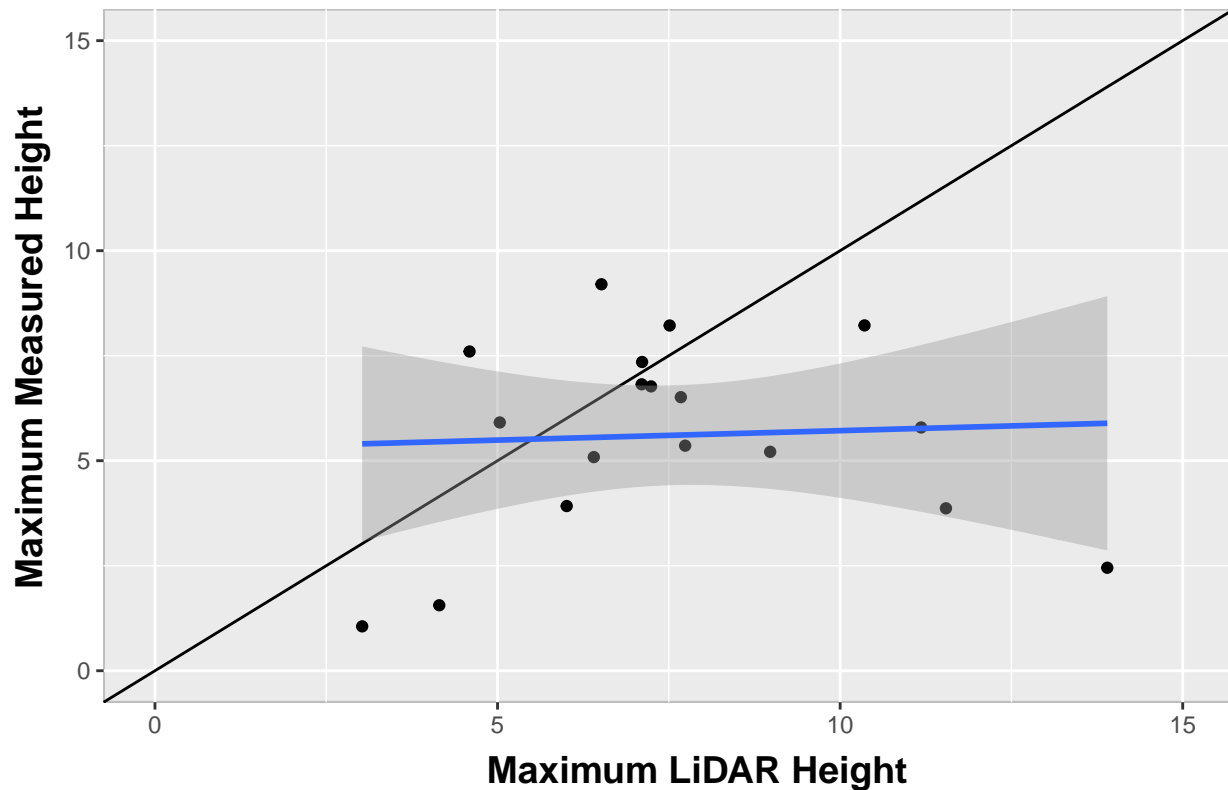


Figure 7: Scatterplot measured height compared to lidar chm.

We can also add a regression fit to our plot. Explore the GGLOT options and customize your plot.

```
plot with regression fit
p <- ggplot(SJER_height@data, aes(x=SJER_lidarCHM, y = insitu_avg)) +
 geom_point() +
 ylab("Maximum Measured Height") +
 xlab("Maximum LiDAR Height")+
 xlim(0,15) + ylim(0,15) + # set x and y limits to 0-20
 geom_abline(intercept = 0, slope=1)+
 geom_smooth(method=lm)

p + theme(panel.background = element_rect(colour = "grey")) +
 ggtitle("LiDAR CHM Derived vs Measured Tree Height") +
 theme(plot.title=element_text(family="sans", face="bold", size=20, vjust=1.9)) +
 theme(axis.title.y = element_text(family="sans", face="bold", size=14, angle=90, hjust=0.54, vjust=1)) +
 theme(axis.title.x = element_text(family="sans", face="bold", size=14, angle=00, hjust=0.54, vjust=-.1))
```

View Difference: lidar vs measured

```
Calculate difference
SJER_height@data$ht_diff <- (SJER_height@data$SJER_lidarCHM - SJER_height@data$insitu_avg)
```

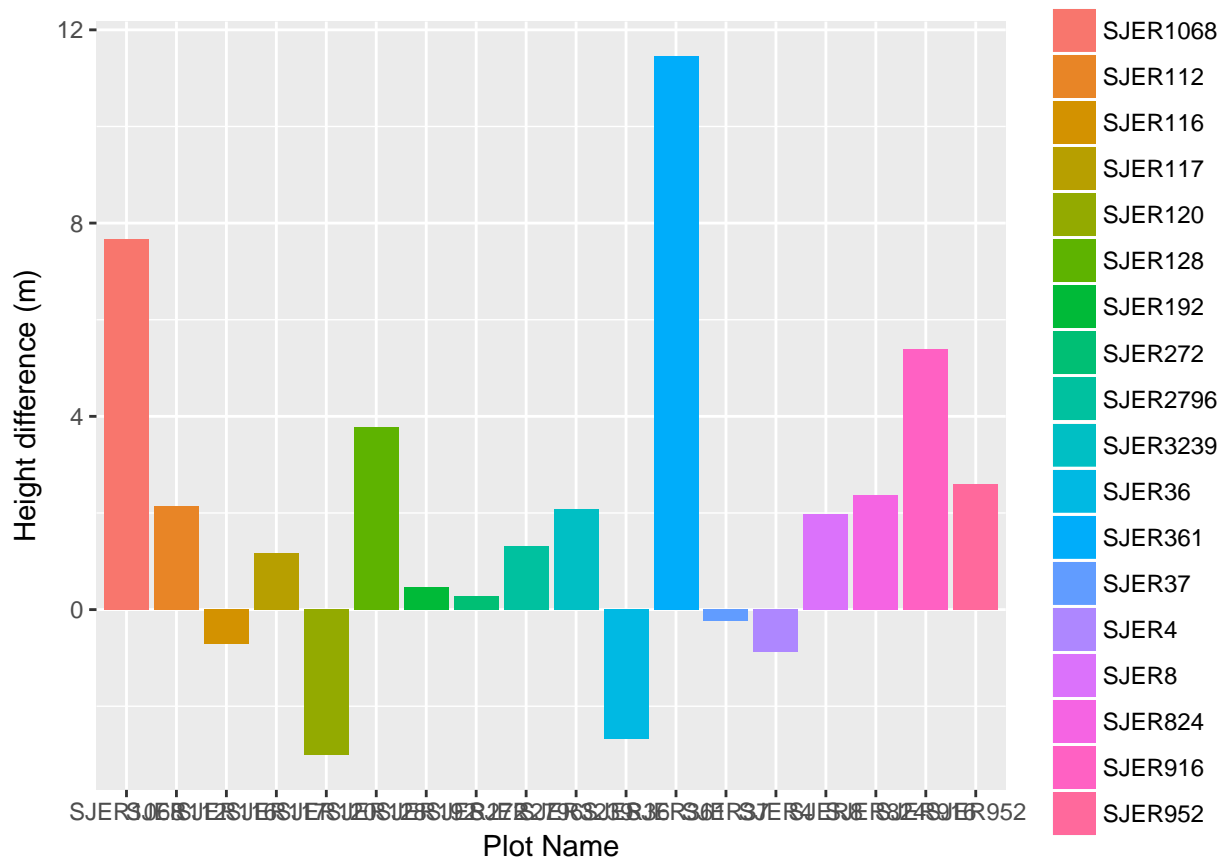


Figure 8: box plot showing differences between chm and measured heights.

```
create bar plot using ggplot()
ggplot(data=SJER_height@data,
 aes(x=Plot_ID, y=ht_diff, fill=Plot_ID)) +
 geom_bar(stat="identity") +
 xlab("Plot Name") + ylab("Height difference (m)")

 ggtitle("Difference: \nLidar avg height - in situ avg height (m)")
$title
[1] "Difference: \nLidar avg height - in situ avg height (m)"
##
$subtitle
NULL
##
attr(,"class")
[1] "labels"
```

You have now successfully created a canopy height model using LiDAR data AND compared LiDAR derived vegetation height, within plots, to actual measured tree height data. Does the relationship look good or not? Would you use lidar data to estimate tree height over larger areas? Would other metrics be a better comparison (see challenge below).

## Test your skills: LiDAR vs Insitu Comparison

Create a plot of LiDAR max height vs *insitu* max height. Add labels to your plot. Customize the colors, fonts and the look of your plot. If you are happy with the outcome, share your plot in the comments below!

### Interactive plot