Raster analysis workflow in R.

Learning Objectives

After completing this tutorial, you will be able to:

- Effectively classify a raster dataset using classes determined by exploring a histogram of the data.

What you need

You will need a computer with internet access to complete this lesson and the data for week 4 of the course.

Download Week 4 Data (~500 MB){:data-proofer-ignore=" .btn }

We can break our data analysis workflow into several steps as follows:

- **Data Processing:** load and "clean" the data. This may include cropping, dealing with NA values, etc
- **Data Exploration:** understand the range and distribution of values in your data. This may involve plotting histograms scatter plots, etc
- **More Data Processing & Analysis:** This may include the final data processing steps that you determined based upon the data exploration phase.
- **Final data analysis:** The final steps of your analysis - often performed using information gathered in the early data processing / exploration stages of your workflow.
- **Presentation:** Refining your results into a final plot or set of plots that are cleaned up, labeled, etc.

Please note - working with data is not a linear process. There are no defined steps. As you work with data more, you will develop your own workflow and approach.

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

# set working directory
setwd("~/Documents/earth-analytics")
options(stringsAsFactors = F)
```

Note: try mapview() is a function that allows you to create interactive maps of spatial data using leaflet as a base.

https://cran.r-project.org/web/packages/mapview/index.html

```
# load data
pre_dtm <- raster("data/week3/BLDR_LeeHill/pre-flood/lidar/pre_DTM.tif")
pre_dsm <- raster("data/week3/BLDR_LeeHill/pre-flood/lidar/pre_DSM.tif")

post_dtm <- raster("data/week3/BLDR_LeeHill/post-flood/lidar/post_DTM.tif")
post_dsm <- raster("data/week3/BLDR_LeeHill/post-flood/lidar/post_DSM.tif")

# import crop extent
crop_ext <- readOGR("data/week3/BLDR_LeeHill", "clip-extent")
## OGR data source with driver: ESRI Shapefile
## Source: "data/week3/BLDR_LeeHill", layer: "clip-extent"
## with 1 features
```

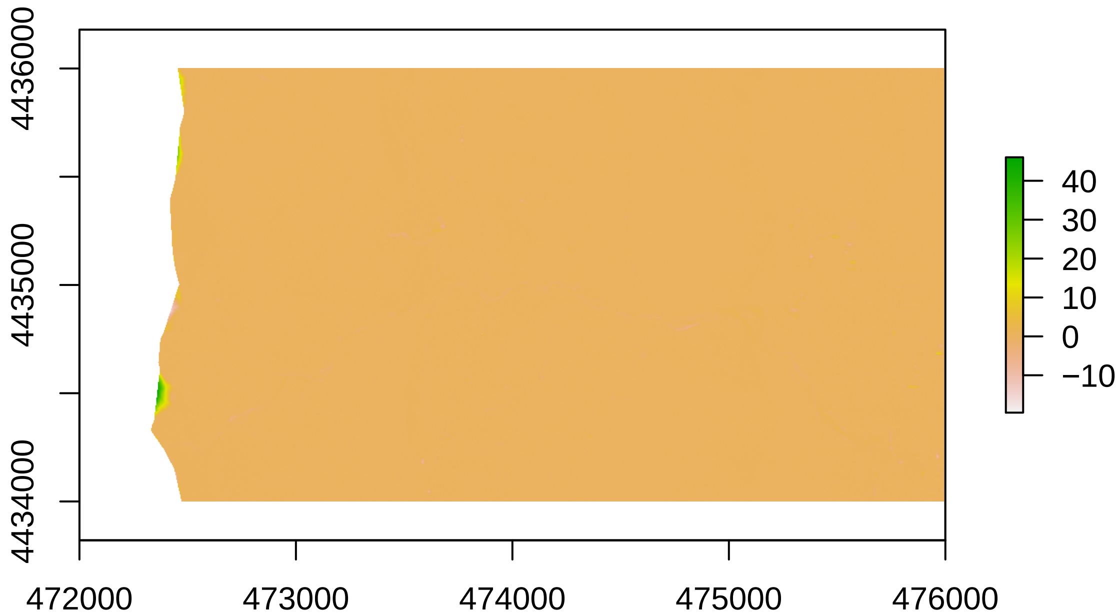

Figure 1: difference between pre and post flood dtm.

```
## It has 1 fields
## Integer64 fields read as strings:  id
```

Calculate the difference.

```r
# calculate dtm difference
dtm_diff_uncropped <- post_dtm - pre_dtm
plot(dtm_diff_uncropped)
```

Next, crop the data.

```r
# crop the data
dtm_diff <- crop(dtm_diff_uncropped, crop_ext)
plot(dtm_diff,
     main="cropped data")
```

```r
# get a quick glimpse at some of the values for a particular "row"
# note there are a LOT of values in this raster so this won't print all values.
# below i used the head() function to limit the n umber of values returned to 6.
# that way a lot of numbers don't print out in my final knitr output.
head(getValues(dtm_diff, row = 5))
## [1]  0.04992676 -0.02001953 -0.10998535 -0.13000488 -0.02001953 -0.20996094

# view max data values
dtm_diff@data@max
## [1] 15.09998
dtm_diff@data@min
## [1] -10.53003
```

```r
# plot histogram of data
hist(dtm_diff,
     main="Distribution of raster cell values in the DTM difference data",
     xlab="Height (m)", ylab="Number of Pixels",
```

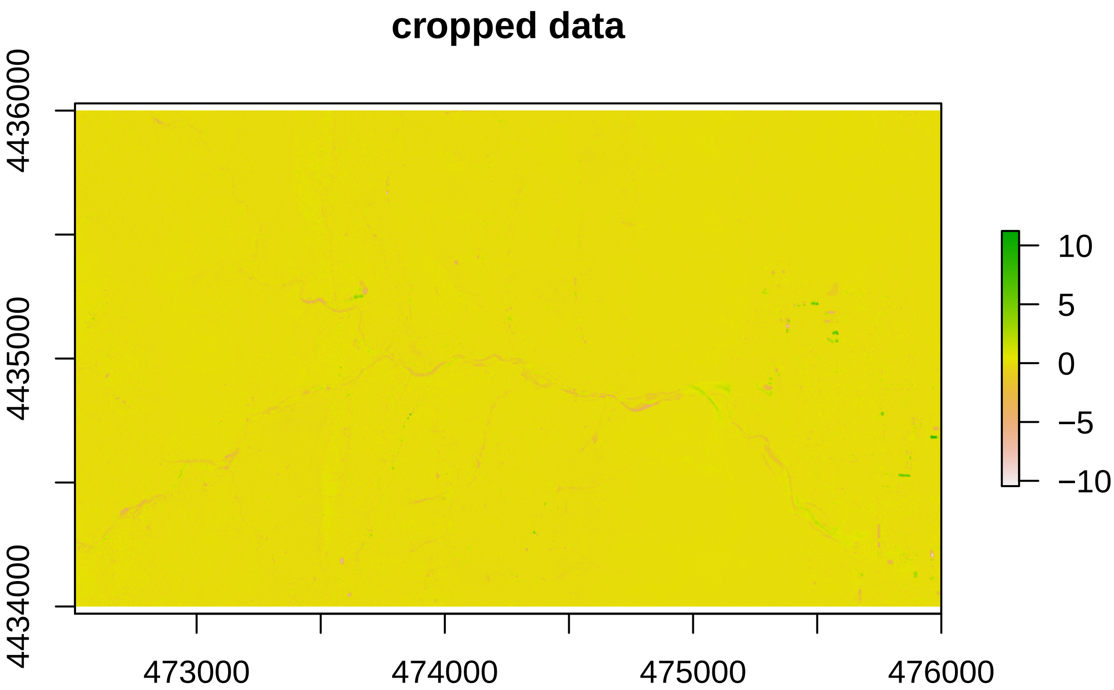

Figure 2: cropped data

```
     col="springgreen")
```

```
hist(dtm_diff,
     xlim=c(-2,2),
     main="Histogram of pre-post flood DTM differences \nZoomed in to -2 to 2 on the x axis",
     col="brown")
```

```
# see how R is breaking up the data
histinfo <- hist(dtm_diff)
```

```
histinfo
## $breaks
##  [1] -11 -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0   1   2   3   4   5
## [18]   6   7   8   9  10  11  12  13  14  15  16
##
## $counts
##  [1]      15      21      65      85     191     306     990    2296
##  [9]    8934   39467 3380797 3522363   18939    3131     883     618
## [17]     524     172      63     111      23       2       2       1
## [25]       0       0       1
##
## $density
##  [1] 2.148997e-06 3.008596e-06 9.312321e-06 1.217765e-05 2.736390e-05
##  [6] 4.383954e-05 1.418338e-04 3.289398e-04 1.279943e-03 5.654298e-03
## [11] 4.843549e-01 5.046365e-01 2.713324e-03 4.485673e-04 1.265043e-04
## [16] 8.853868e-05 7.507163e-05 2.464183e-05 9.025788e-06 1.590258e-05
## [21] 3.295129e-06 2.865330e-07 2.865330e-07 1.432665e-07 0.000000e+00
## [26] 0.000000e+00 1.432665e-07
##
```

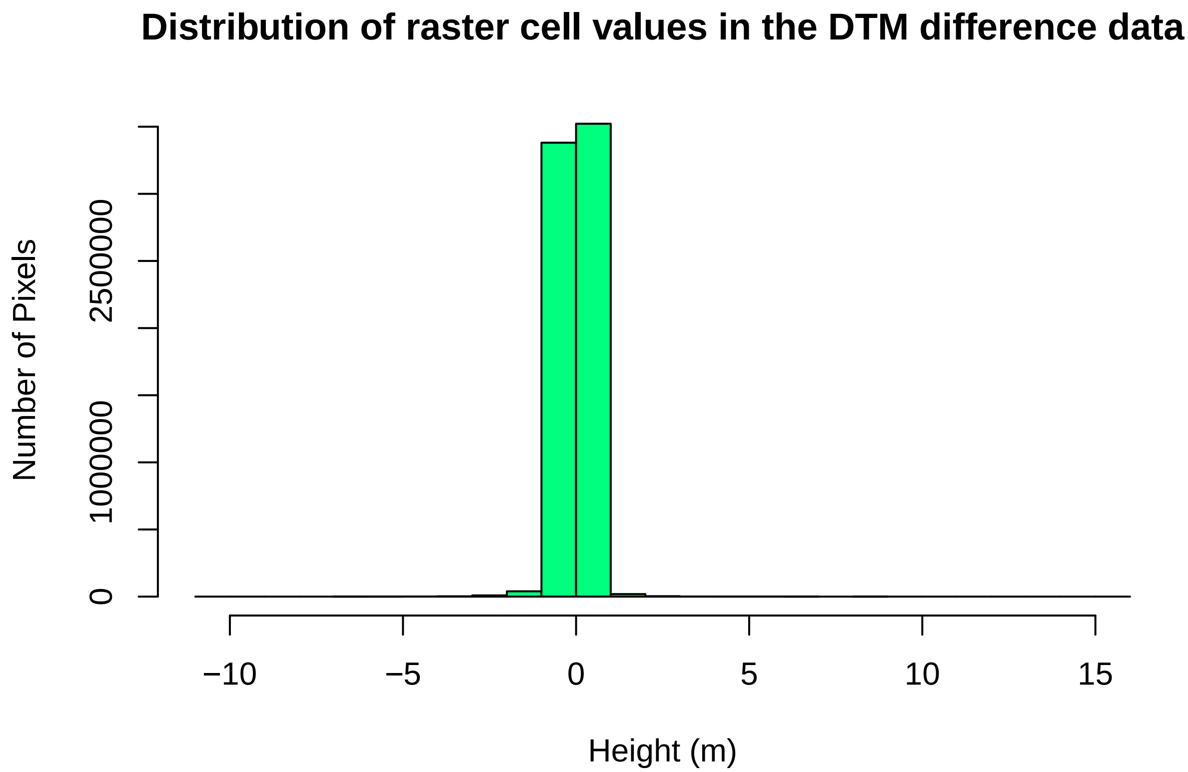

Figure 3: initial histogram

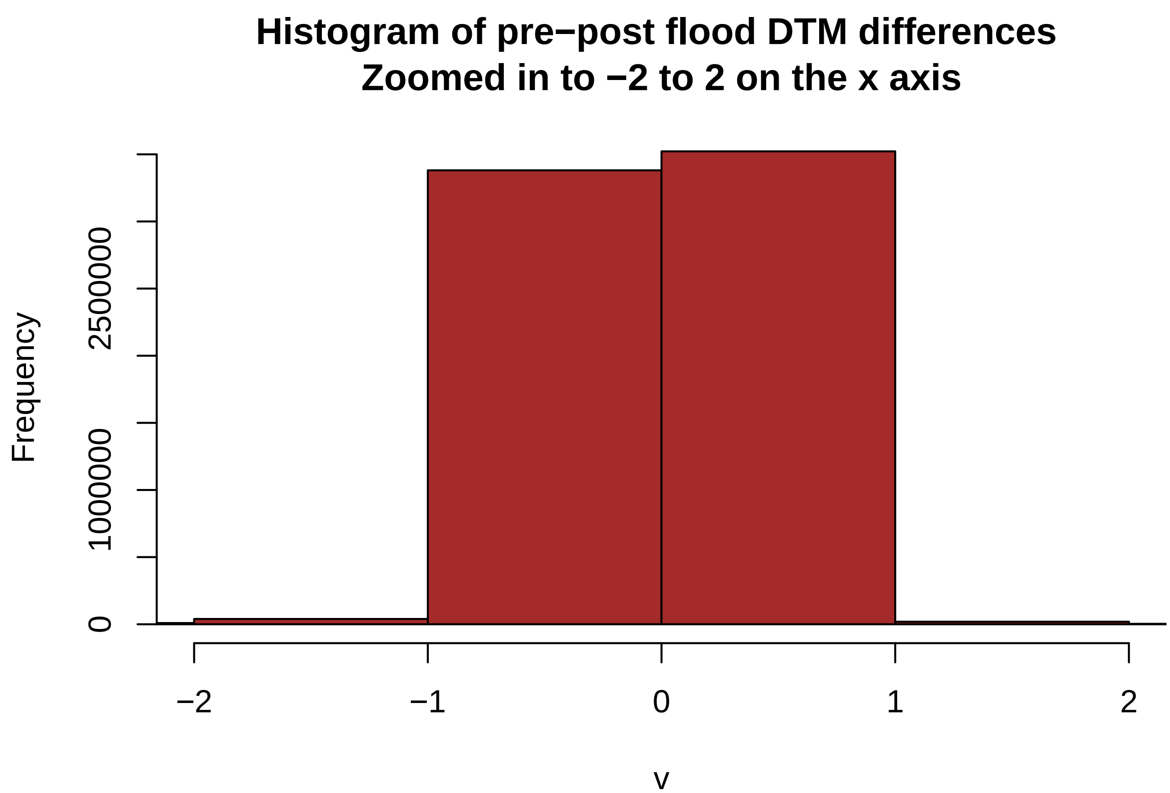

Figure 4: initial histogram w xlim to zoom in

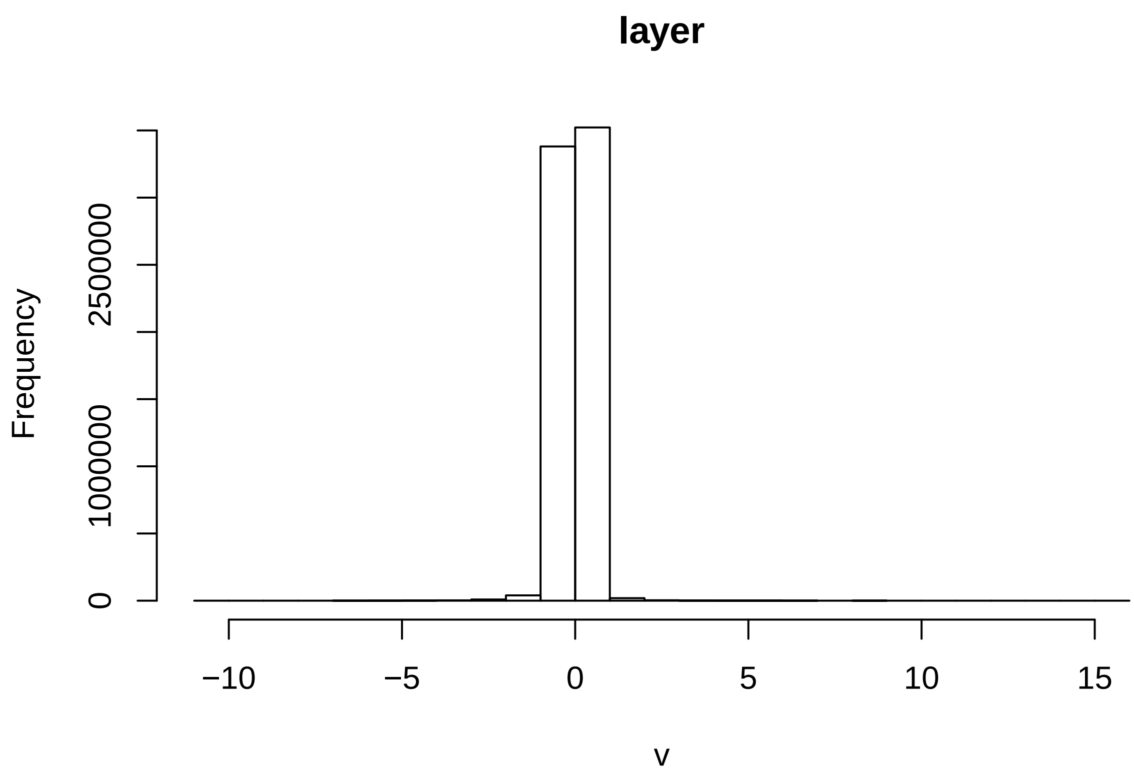

Figure 5: initial histogram w xlim to zoom in

```
## $mids
##  [1] -10.5  -9.5  -8.5  -7.5  -6.5  -5.5  -4.5  -3.5  -2.5  -1.5  -0.5
## [12]   0.5   1.5   2.5   3.5   4.5   5.5   6.5   7.5   8.5   9.5  10.5
## [23]  11.5  12.5  13.5  14.5  15.5
##
## $xname
## [1] "v"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"

# how many breaks does R use in the default histogram
length(histinfo$breaks)
## [1] 28

# summarize values in the data
summary(dtm_diff, na.rm=T)
##                 layer
## Min.    -10.53002930
## 1st Qu.  -0.06994629
## Median    0.01000977
## 3rd Qu.   0.07995605
## Max.     15.09997559
```

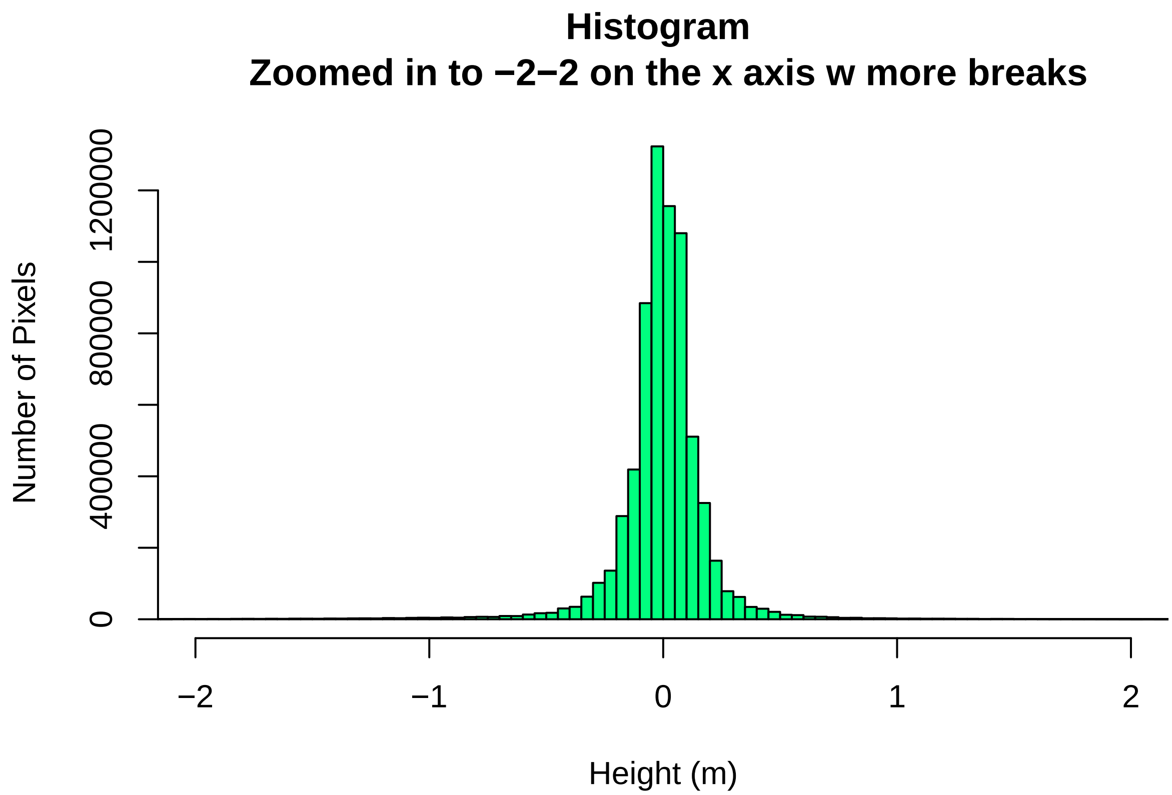

Figure 6: initial histogram w xlim to zoom in and breaks

```
## NA's       0.00000000
```

Breaks

Above, we saw that we can see how R breaks up our data to create a histogram. R, by default, creates 35 bins to plot a histogram of our raster data. We can increase the number of breaks or bins that the hist0gram uses with the argument:

`breaks=number`

In the example below, I used a very large number - 500 so we can see the bins.

```
# where are most of our values?
hist(dtm_diff,
     xlim=c(-2,2),
     breaks=500,
     main="Histogram \n Zoomed in to -2-2 on the x axis w more breaks",
     xlab="Height (m)", ylab="Number of Pixels",
     col="springgreen")
```

Histogram with custom breaks

We can create custom breaks or bins in a histogram too. To do this, we pass the same `breaks` argument a vector of numbers that represent the range for each bin in our histogram.

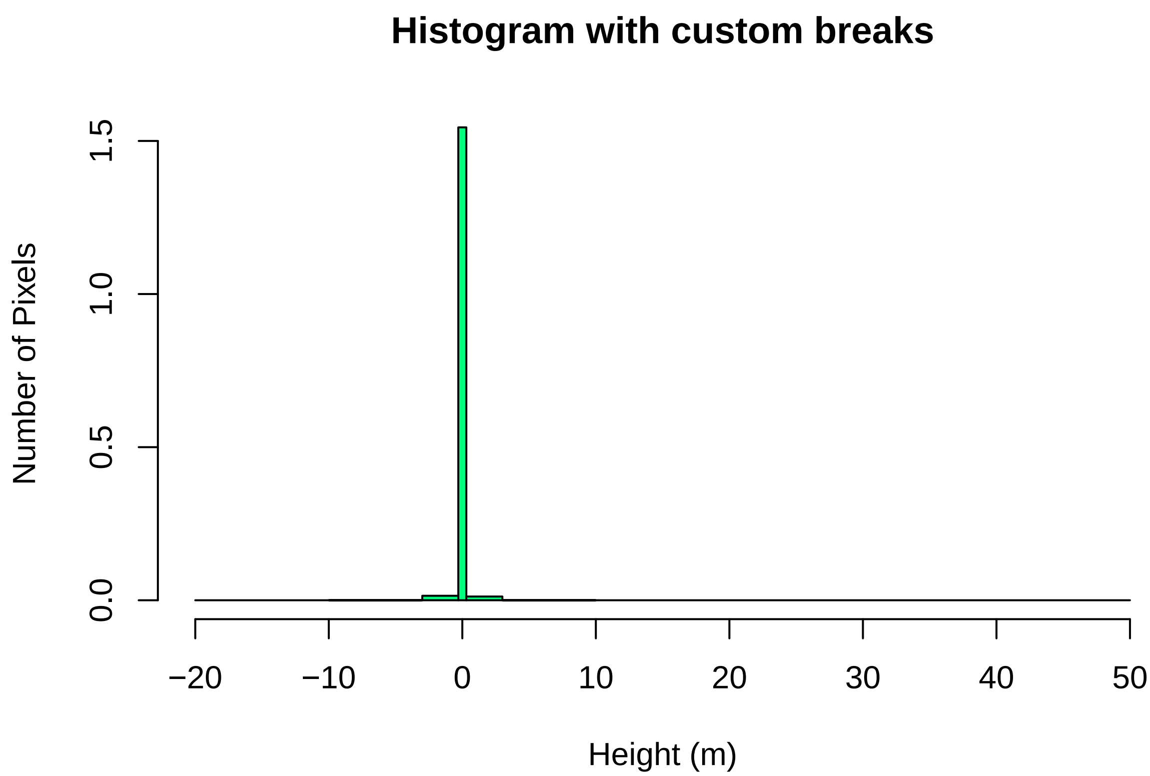

Figure 7: histogram w custom breaks

```r
# We may want to explore breaks in our histogram before plotting our data
hist(dtm_diff,
     breaks=c(-20, -10, -3, -.3, .3, 3, 10, 50),
     main="Histogram with custom breaks",
     xlab="Height (m)" , ylab="Number of Pixels",
     col="springgreen")
```

Finally, let's plot the data using the breaks that we created for our histogram above. We know that there is a high number of cells with a value between -1 and 1. So let's consider that when we select the colors for our plot.

```r
# plot dtm difference with breaks
plot(dtm_diff,
     breaks=c(-20, -10, -3, -.3, .3, 3, 10, 50),
     col=terrain.colors(7),
     main="DTM Difference \n Using manual breaks")
```

Custom plot colors

Next, let's adjust the colors that we use to plot our raster. to do that we will create a vector of colors, each or which will represent one of our numeric "bins" of raster values.

This mimics a classified map - we are still exploring our data!

```r
# how many breaks do we have?
# NOTE: we will have one less break than the length of this vector
```

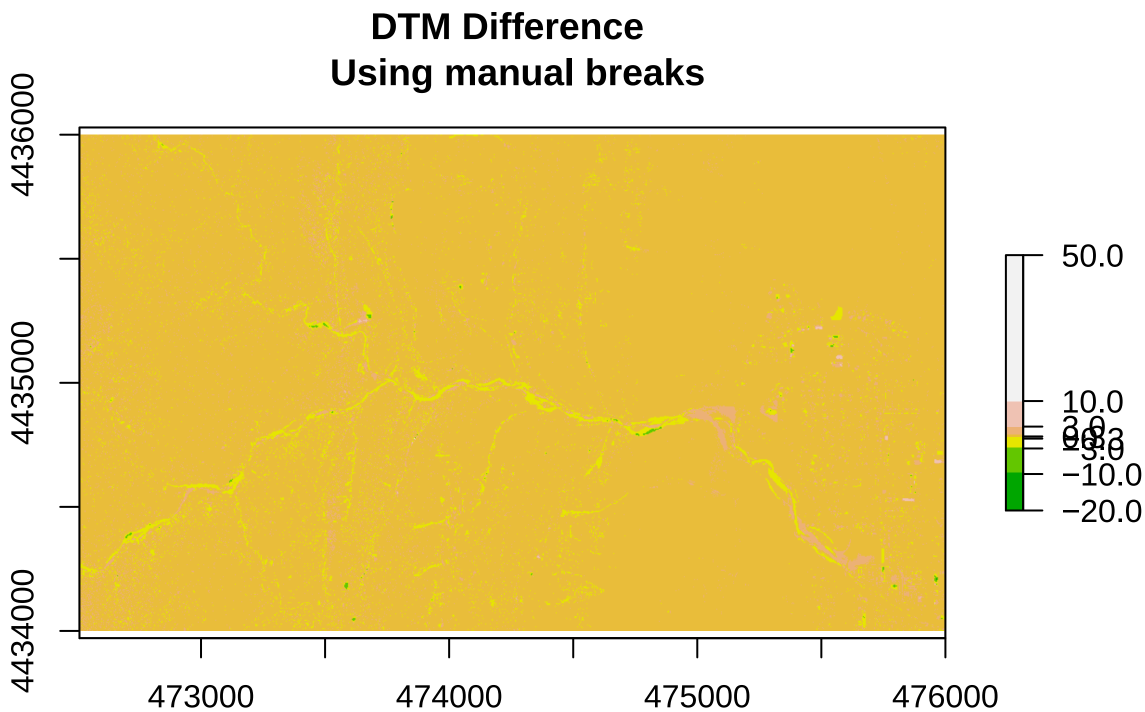

Figure 8: Plot difference dtm.

```r
length(c(-20,-10,-3,-1, 1, 3, 10, 50))
## [1] 8
```

Set number of colors based upon how many breaks or bins we have in our data above we have 8 numbers in our breaks vector. this translates to 7 bins each or which requires a unique color.

```r
# create a vector of colors - one for each "bin" of raster cells
diff_colors <- c("palevioletred4", "palevioletred1", "ivory1",
                 "seagreen1", "seagreen4")
plot(dtm_diff,
     breaks=c(-20, -3, -.3, .3, 3, 50),
     col=diff_colors,
     legend=F,
     main="Plot of DTM differences\n custom colors & manual breaks")

# make sure legend plots outside of the plot area
par(xpd=T)
# add the legend to the plot
legend(x=dtm_diff@extent@xmax, y=dtm_diff@extent@ymax, # legend location
       legend=c("-20 to -3", "-3 to -.3",
                "-.3 to .3", ".3 to 3", "3 to 50"),
       fill=diff_colors,
       bty="n",
       cex=.7)
```

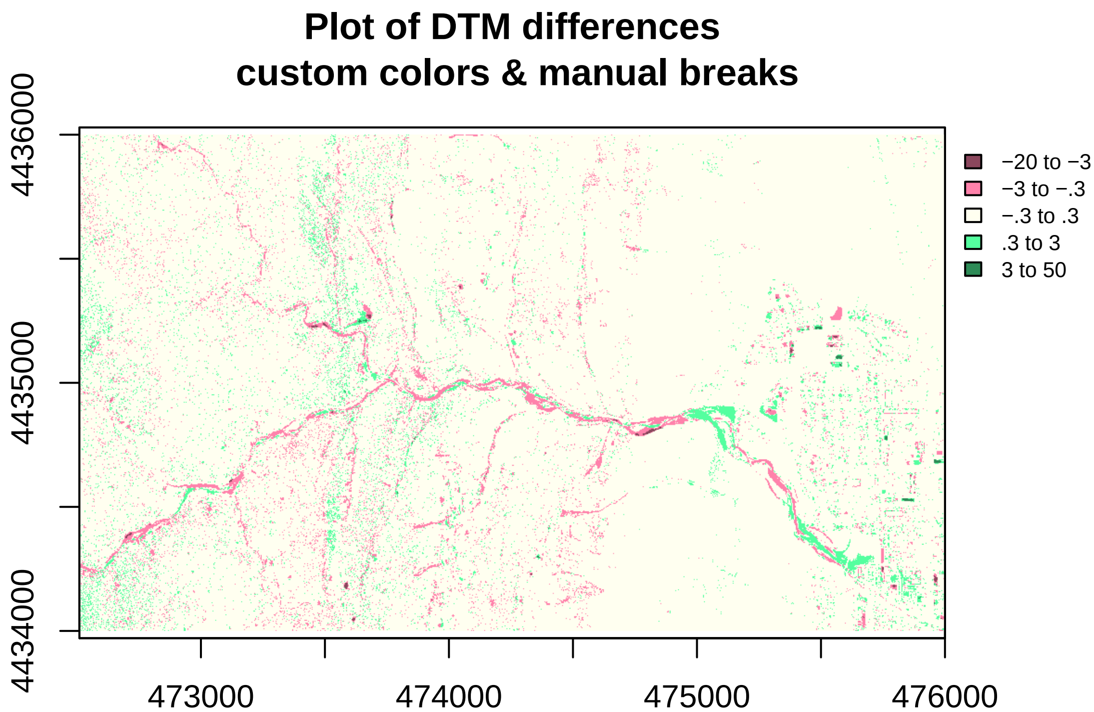

Figure 9: Plot difference dtm with custom colors.

Crop and replot

We can zoom into a part of the raster manually - by first cropping the data using a manually created plot extent. Then plotting the newly cropped raster subset.

```
# new_extent <- drawExtent()
new_extent <- extent(473690, 474155.2, 4434849, 4435204)
new_extent
## class       : Extent
## xmin        : 473690
## xmax        : 474155.2
## ymin        : 4434849
## ymax        : 4435204

# crop the raster to a smaller area
dtm_diff_crop <- crop(dtm_diff, new_extent)

# Plot the cropped raster
plot(dtm_diff_crop,
     breaks=c(-20, -3, -.3, .3, 3, 50),
     col=diff_colors,
     legend=F,
     main="Lidar DTM Difference \n cropped subset")

# grab the upper right hand corner coordinates to place the legend.
legendx <- dtm_diff_crop@extent@xmax
legendy <- dtm_diff_crop@extent@ymax
```

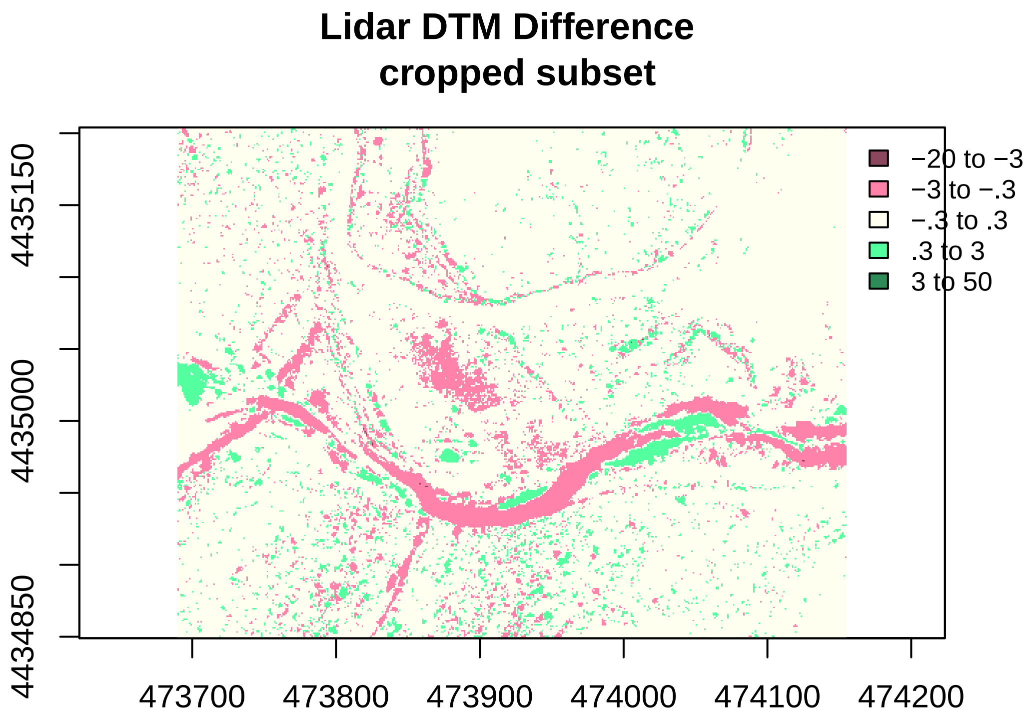

Figure 10: cropped dtm subset

```
par(xpd=TRUE)
legend(legendx, legendy,
       legend=c("-20 to -3", "-3 to -.3", "-.3 to .3",
                ".3 to 3", "3 to 50"),
       fill=diff_colors,
       bty="n",
       cex=.8)
```

```
dev.off()
## RStudioGD
##         2
```

Create a final classified dataset

When we have decided what break points work best for our data, then we may chose to classify the data.

```
# create reclass vector
reclass_vector <- c(-20,-3, -2,
                    -3, -.3, -1,
                    -.3, .3, 0,
                    .3, 3, 1,
                    3, 50, 2)

reclass_matrix <- matrix(reclass_vector,
                         ncol=3,
                         byrow = T)
```

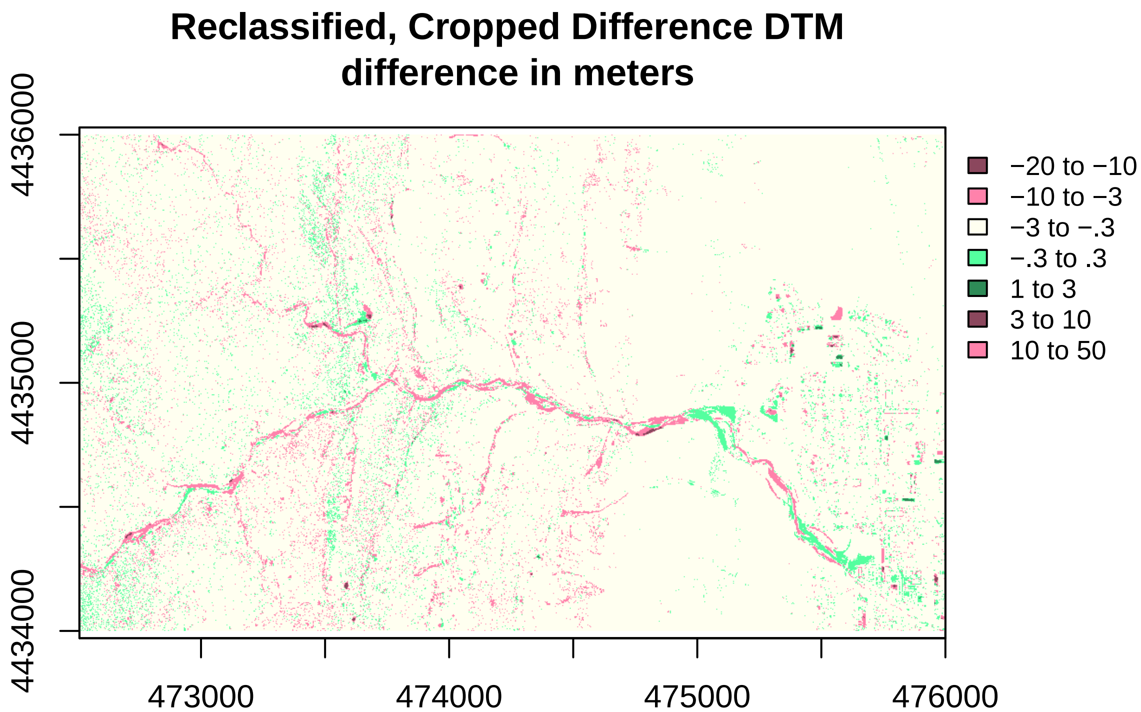

Figure 11: final plot

```
reclass_matrix
##       [,1] [,2] [,3]
## [1,] -20.0 -3.0   -2
## [2,]  -3.0 -0.3   -1
## [3,]  -0.3  0.3    0
## [4,]   0.3  3.0    1
## [5,]   3.0 50.0    2
```

Reclassify difference raster

```
diff_dtm_rcl <- reclassify(dtm_diff, reclass_matrix)

plot(diff_dtm_rcl,
     col=diff_colors,
     legend=F,
     main="Reclassified, Cropped Difference DTM \n difference in meters")
par(xpd=T)
legend(dtm_diff@extent@xmax, dtm_diff@extent@ymax,
       legend=c("-20 to -10", "-10 to -3", "-3 to -.3",
                "-.3 to .3", "1 to 3", "3 to 10", "10 to 50"),
       fill=diff_colors,
       bty="n",
       cex=.8)
```

Finally view the final histogram

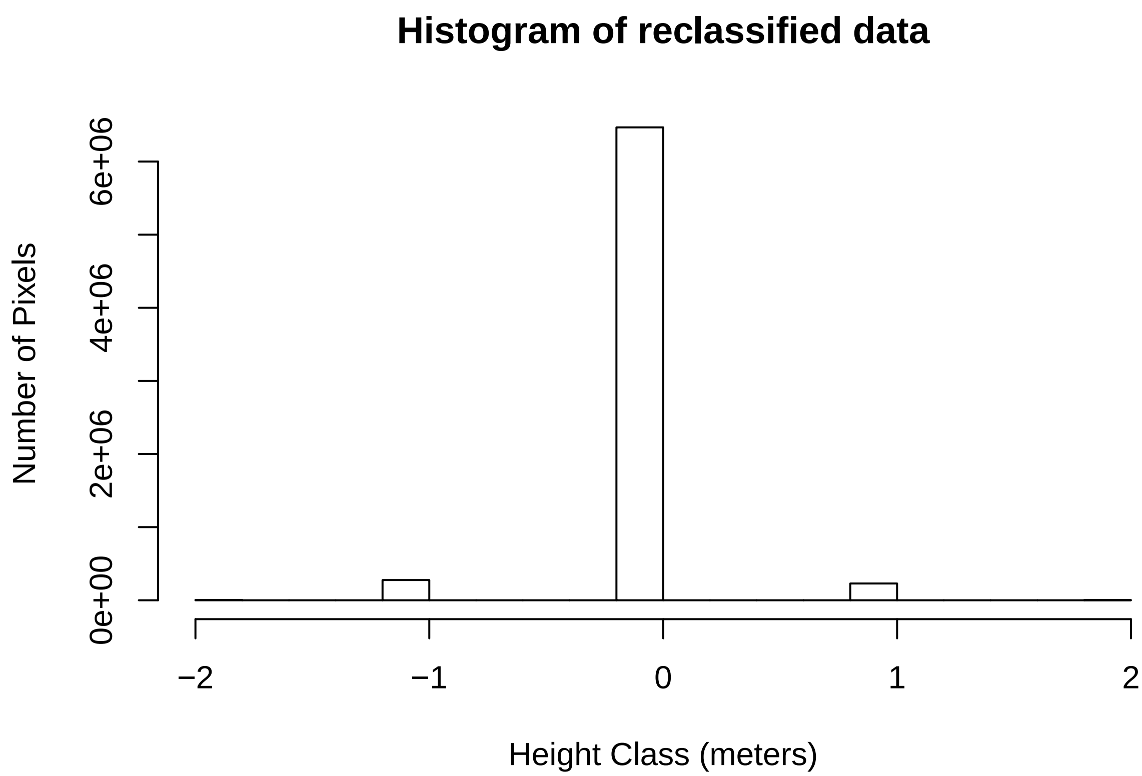

Figure 12: histogram of differences

```
hist(diff_dtm_rcl,
     main="Histogram of reclassified data",
     xlab="Height Class (meters)",
     ylab="Number of Pixels")
```

The above histogram looks odd. This is because we are trying to force our discrete data into bins. A barplot is a more appropriate plot.

```
barplot(diff_dtm_rcl,
     main="Barplot of reclassified data",
     xlab="Height Class (meters)",
     ylab="Frequency of Pixels",
     col=diff_colors)
## Warning in .local(height, ...): a sample of 14.3% of the raster cells were
## used to estimate frequencies
```

Now let's look at one last thing. What would the distribution look like if we set all values between -.3 to .3 to NA?

```
# create a new raster object
diff_dtm_rcl_na <- diff_dtm_rcl
# assign values between -.3 and .3 to NA
diff_dtm_rcl_na[diff_dtm_rcl_na >= -.3 & diff_dtm_rcl_na <= .3] <- NA
# view histogram
barplot(diff_dtm_rcl_na,
     main="Barplot of data \n values between -.3 and .3 set to NA",
     xlab="Difference Class",
```

Figure 13: histogram of differences

```
     col=diff_colors)
## Warning in .local(height, ...): a sample of 14.3% of the raster cells were
## used to estimate frequencies
```

```
# view summary of data
summary(diff_dtm_rcl_na)
##            layer
## Min.          -2
## 1st Qu.       -1
## Median        -1
## 3rd Qu.        1
## Max.           2
## NA's    6467991
```

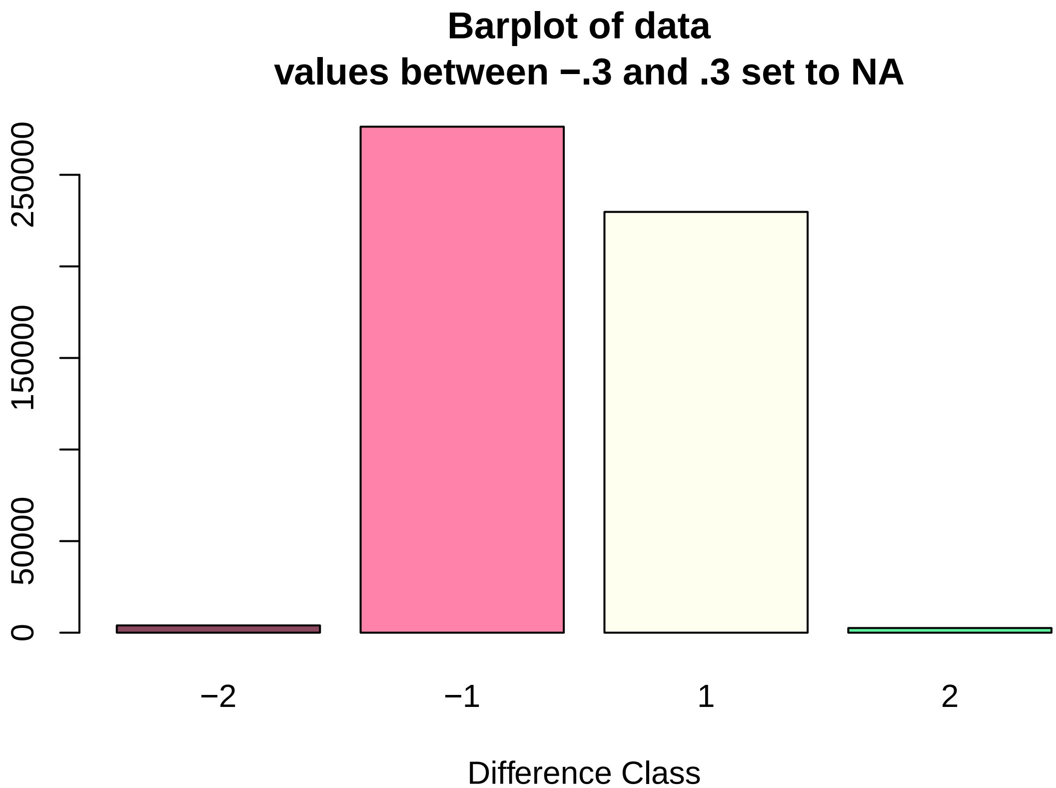

Figure 14: histogram of final cleaned data