SOMs_RS

Simone

9/22/2022

Import raster

Import single band and 3-bands rasters

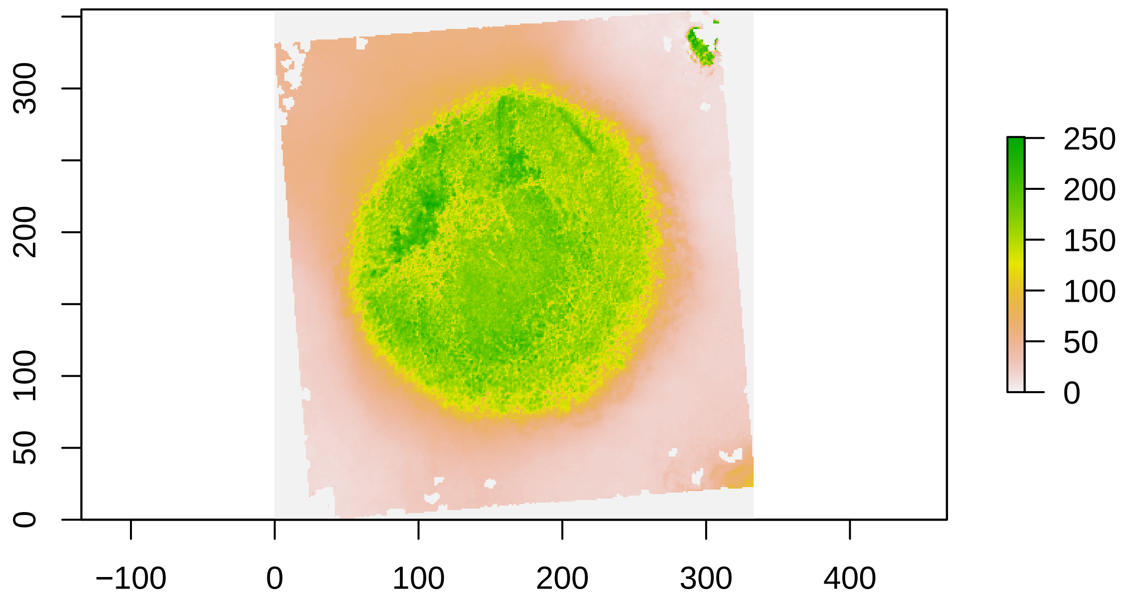
```
imp.reef.sb <- raster("D:/HIMB/MBIO630/Data/Reef_20_2022_01_18_20cm_pix.tif")
print(imp.reef.sb)
```

```
## class      : RasterLayer
## band       : 1  (of  4  bands)
## dimensions : 355, 333, 118215  (nrow, ncol, ncell)
## resolution : 1, 1  (x, y)
## extent     : 0, 333, 0, 355  (xmin, xmax, ymin, ymax)
## crs        : NA
## source     : Reef_20_2022_01_18_20cm_pix.tif
## names      : Reef_20_2022_01_18_20cm_pix
## values     : 0, 255  (min, max)
```

```
imp.reef <- raster::brick("D:/HIMB/MBIO630/Data/Reef_20_2022_01_18_20cm_pix.tif")
print(imp.reef)
```

```
## class      : RasterBrick
## dimensions : 355, 333, 118215, 4  (nrow, ncol, ncell, nlayers)
## resolution : 1, 1  (x, y)
## extent     : 0, 333, 0, 355  (xmin, xmax, ymin, ymax)
## crs        : NA
## source     : Reef_20_2022_01_18_20cm_pix.tif
## names      : Reef_20_2022_01_18_20cm_pix.1, Reef_20_2022_01_18_20cm_pix.2, Reef_20_2022_01_18_20cm_pi
## min values :                             0,                             0,
## max values :                           255,                           255,
```

```
plot(imp.reef.sb)
```



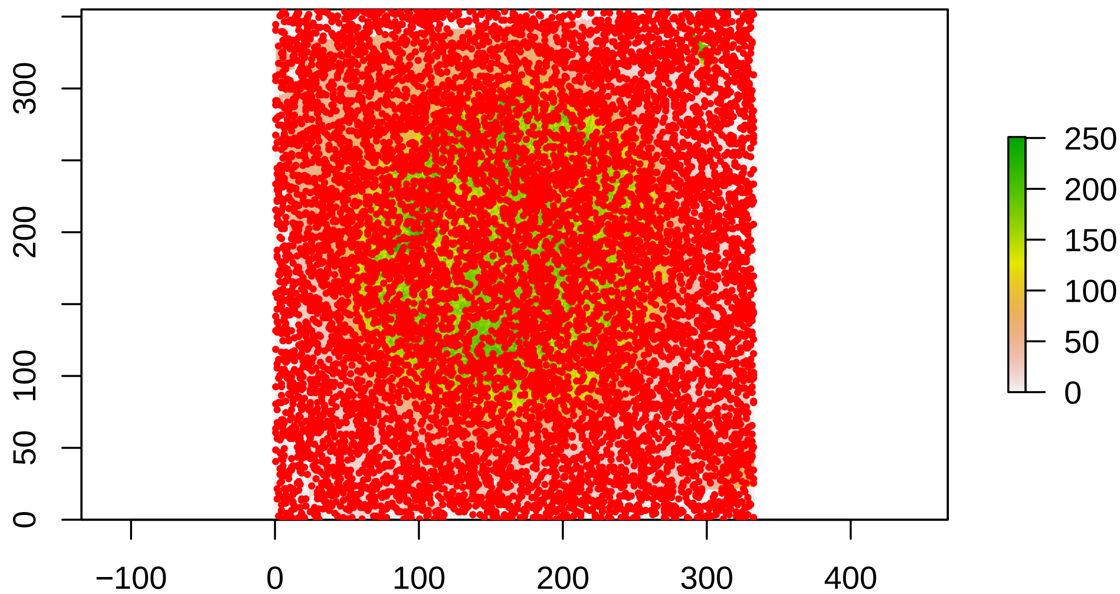
Sampling RGB data from random sample

Generate a number of random points for creating the training dataset.

```
sample.size <- 10000
sample.values <- sample(1:length(imp.reef.sb), sample.size, replace = F)
```

And extract geoinformation.

```
sample.coor <- as.data.frame(xyFromCell(imp.reef, sample.values))
colnames(sample.coor) <- c("Lon", "Lat")
plot(imp.reef.sb)
points(sample.coor$Lon, sample.coor$Lat, col = "red", pch = 16, cex = 0.5)
```



Generate dataframe of selected RGB vectors

```
sample.rgb <- as.data.frame(matrix(nrow = sample.size, ncol = 3))
colnames(sample.rgb) <- c('R', 'G', 'B')
sample.rgb$R <- imp.reef$Reef_20_2022_01_18_20cm_pix.1[sample.values]
sample.rgb$G <- imp.reef$Reef_20_2022_01_18_20cm_pix.2[sample.values]
sample.rgb$B <- imp.reef$Reef_20_2022_01_18_20cm_pix.3[sample.values]
```

Train the Self-Organizing Map

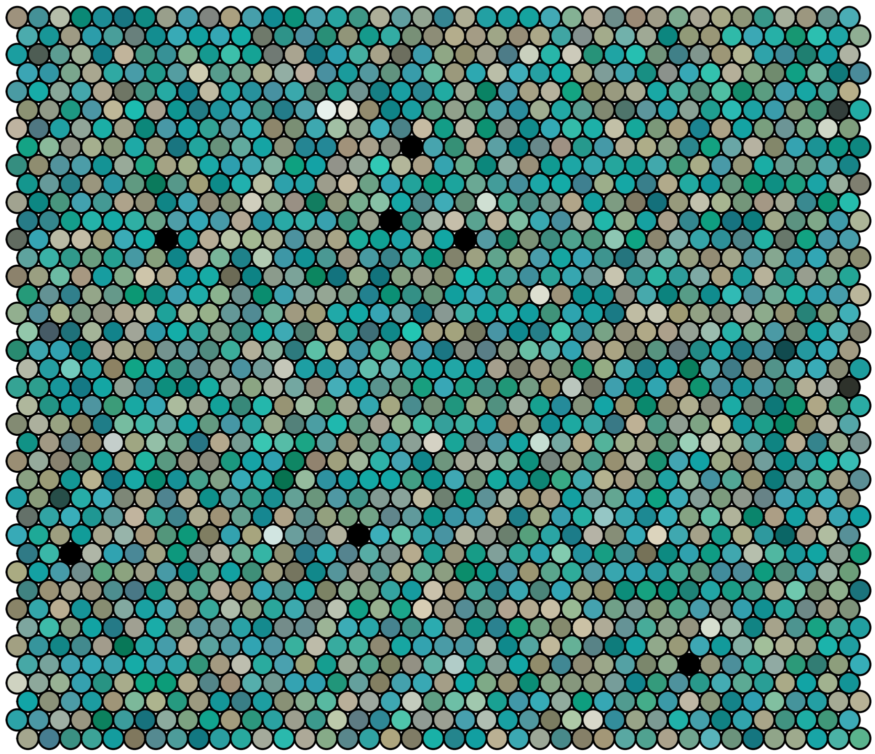
Define a grid for the SOM and extract some data to make it easier to use.

```
grid.size <- ceiling(sample.size ^ (1/2.5))
som.grid <- somgrid(xdim = grid.size, ydim = grid.size, topo = 'hexagonal')
som.model <- kohonen2::som(data.matrix(sample.rgb), grid = som.grid, toroidal = TRUE)
som.events <- som.model$codes
som.events.colors <- rgb(som.events[,1], som.events[,2], som.events[,3], maxColorValue = 255)
som.dist <- as.matrix(dist(som.events))
col.func <- colorRampPalette(c("grey","forestgreen", "darkolivegreen1", "orange"))
```

Generate a plot of the untrained data. this isn't really the configuration at first iteration, but serves as an example.

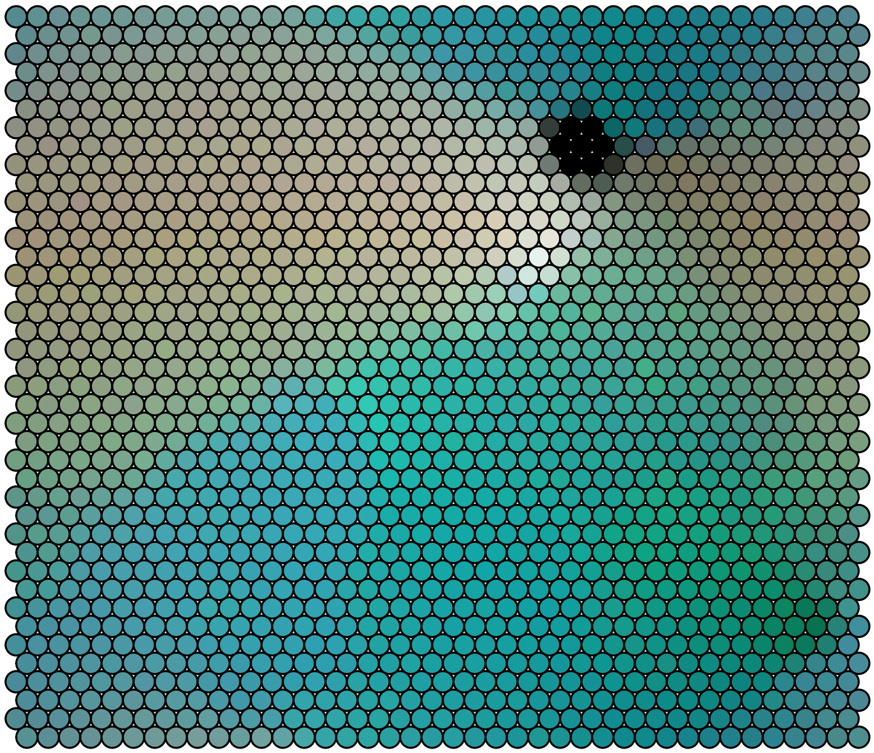
```
plot(som.model,
     type = 'mapping',
```

```
     bgcol = som.events.colors[sample.int(length(som.events.colors), size = length(som.events.colors))]
     keepMargins = F,
     col = NA,
     main = '')
```



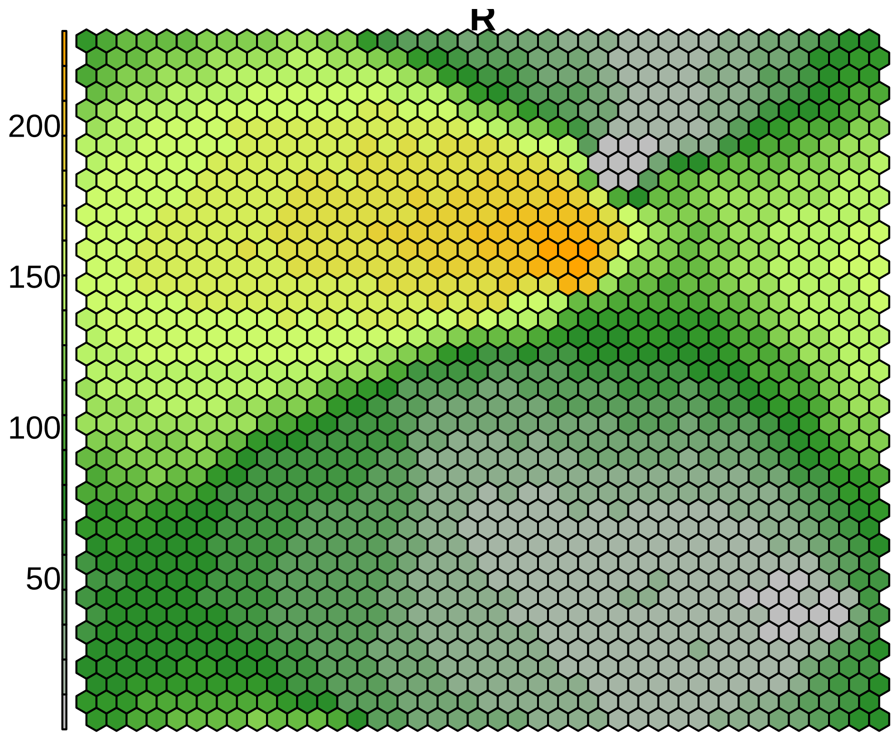
And compare it with the trained model.

```
plot(som.model,
     type = 'mapping',
     bg = som.events.colors,
     keepMargins = F,
     col = NA,
     main = '')
```

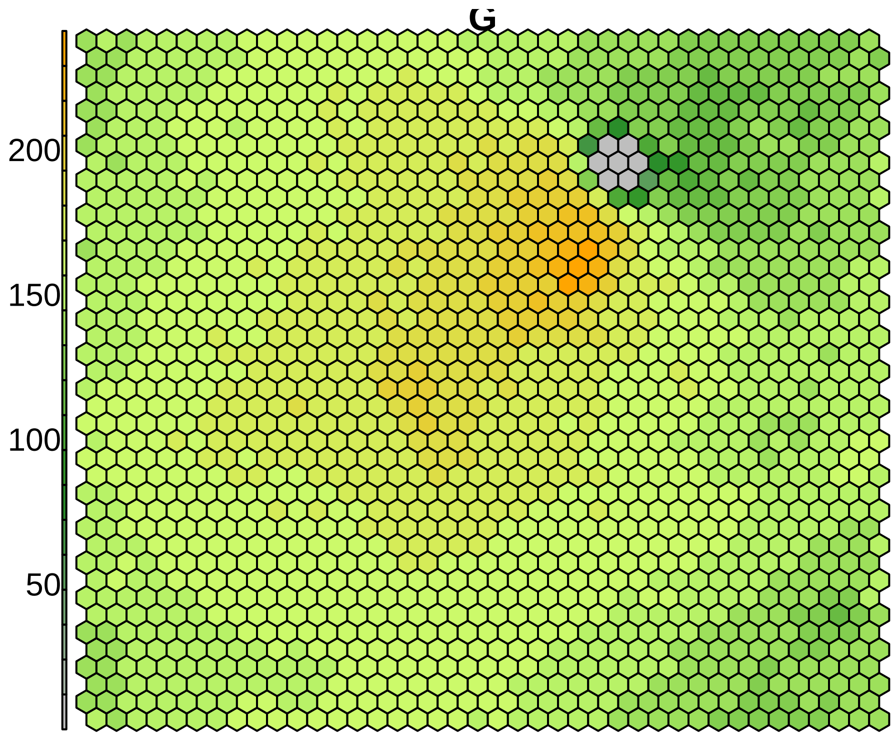


Lets take a look to the RGB values distribution on the SOM (in order: RED, GREEN, BLUE).

```
plot.kohonen(som.model, type = "property", property = som.model$codes[,1],
             main = colnames(sample.rgb)[1],
             palette.name = col.func)
```




```
plot.kohonen(som.model, type = "property", property = som.model$codes[,2],
             main = colnames(sample.rgb)[2],
             palette.name = col.func)
```

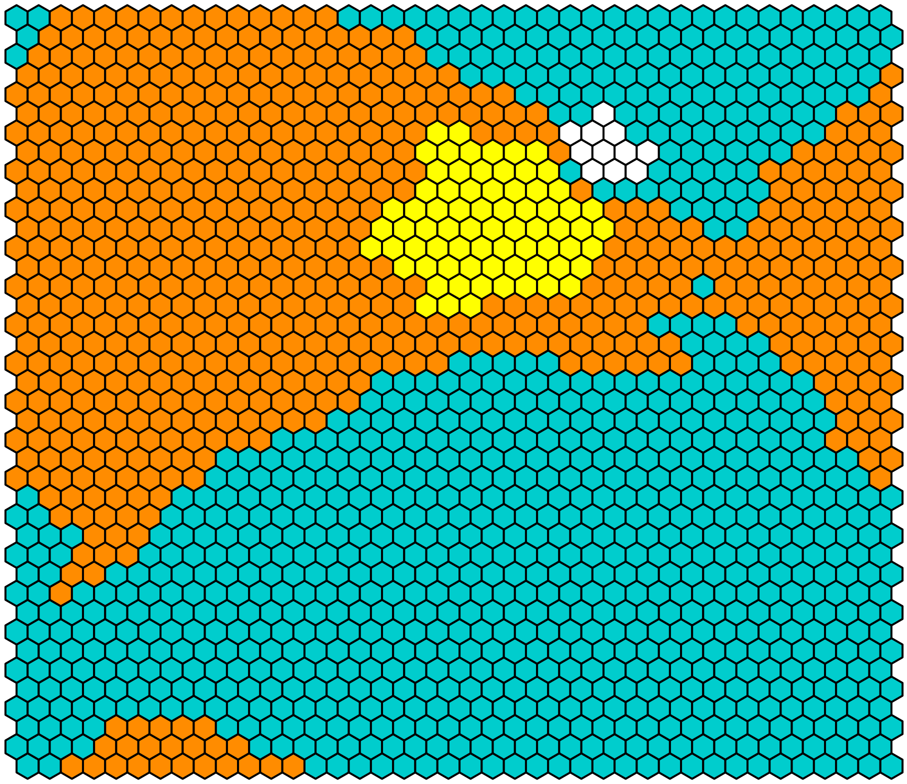



```r
plot.kohonen(som.model, type = "property", property = som.model$codes[,2],
             main = colnames(sample.rgb)[2],
             palette.name = col.func)
```

Clustering

Define colors and clusters (search "Average Silohuette Method for optimizing number of clusters K).

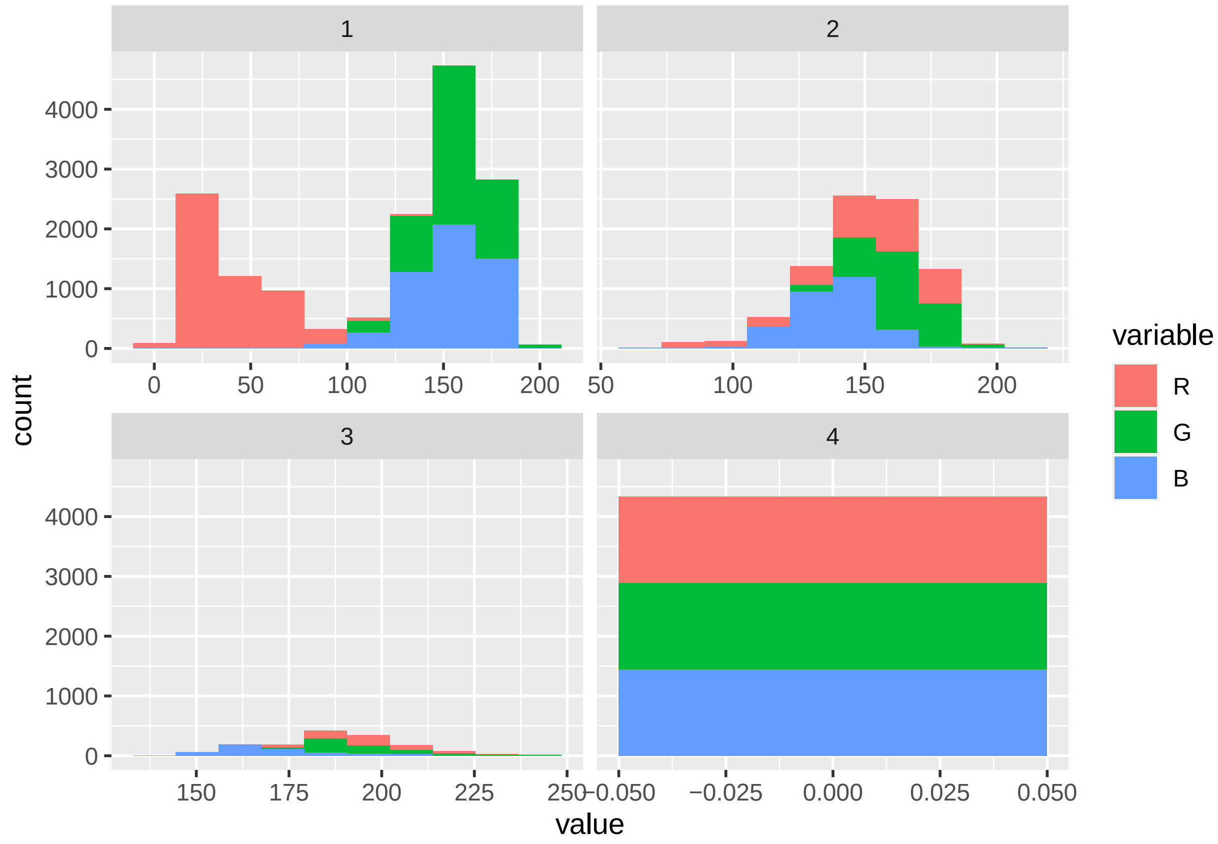
```r
myColor <- c("cyan3","darkorange", "yellow", "white")
col.func <- colorRampPalette(c("grey","forestgreen", "darkolivegreen1", "orange"))
som_cluster <- cutree(hclust(dist(som.model$codes)), 4)
bgcolors <- myColor[som_cluster]
plot.kohonen(som.model,
     type = 'mapping',
     keepMargins = F,
     col = NA,
     main = '')
```



Plot histograms showing RGB values for each class

```
sample.rgb.classes <- som_cluster[som.model$unit.classif]
sample.rgb$CLASS <- sample.rgb.classes
melt.sample.rgb <- melt(sample.rgb, id = c("CLASS"))

ggplot(melt.sample.rgb, aes(value, fill = variable)) +
  geom_histogram(bins = 10) +
  facet_wrap(~CLASS, scales = 'free_x')
```



Predicting new classes

Create input dataset

```
all.cells <- 1:length(imp.reef.sb)
pred.cells <- all.cells[-sample.values]
new.data <- as.data.frame(matrix(nrow = length(imp.reef.sb)-sample.size, ncol = 3))
colnames(new.data) <- c('R', 'G', 'B')
new.data$R <- imp.reef$Reef_20_2022_01_18_20cm_pix.1[pred.cells]
new.data$G <- imp.reef$Reef_20_2022_01_18_20cm_pix.2[pred.cells]
new.data$B <- imp.reef$Reef_20_2022_01_18_20cm_pix.3[pred.cells]
new.data.units <- map.kohonen(som.model, newdata = data.matrix(new.data))
sample.data.units <- map.kohonen(som.model, newdata = data.matrix(sample.rgb))
```

Get the classification for closest map units

```
new.data.classes <- som_cluster[new.data.units$unit.classif]
sample.data.classes <- som_cluster[sample.data.units$unit.classif]
class.mat <- matrix(NA, nrow = length(imp.reef.sb), ncol = 4)
class.mat[sample.values, 1] <-  sample.rgb$R
class.mat[-sample.values, 1] <- new.data$R
class.mat[sample.values, 2] <-  sample.rgb$G
class.mat[-sample.values, 2] <- new.data$G
class.mat[sample.values, 3] <-  sample.rgb$B
```

```r
class.mat[-sample.values, 3] <- new.data$B
class.mat[sample.values, 4] <-  sample.data.classes
class.mat[-sample.values, 4] <- new.data.classes
```

Plotting predicted classes on my raster

```r
pred.reef <- imp.reef
pred.reef$Reef_20_2022_01_18_20cm_pix.4<- class.mat[ ,4]

plot(pred.reef$Reef_20_2022_01_18_20cm_pix.4, breaks=c(0, 1, 2, 3, 4),
     col = myColor)
```