R Workshop week 9: color palette resources

Teal Potter

12/10/2021

Table of Contents

Setup 1

Premade palettes EASIEST 3

3 resources for finding & making custom palettes 4

1. Colour Picker, a point-and-click addin for R Studio EASIEST 4
2. Colorbrewer2.org, a simple webpage INTERMEDIATE 5
3. Coolers, a comprehensive website ADVANCED 6

Apply your palette to a graph 8

Make and save custom theme 8

More info on colorblind visualizations 10

Greyscale graphs 10

Applying continuous scale palettes 12

Here are some resources to help you efficiently choose colors for your graphs and apply them.

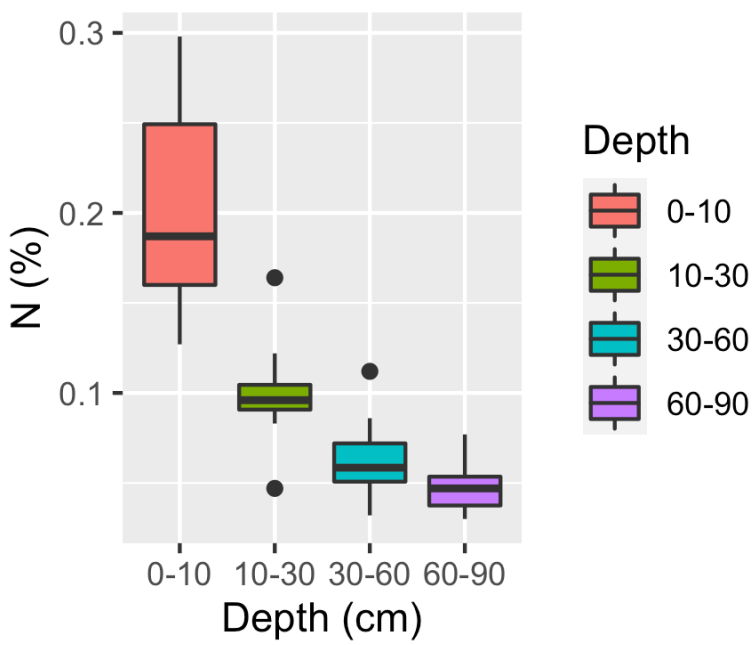
Setup

```
library(ggplot2)
library(car)
library(RColorBrewer)
library(colourpicker)
library(ggsci)
```

```
head(Soils) # our example dataset

##   Group Contour Depth Gp Block   pH     N Dens   P    Ca   Mg    K   Na Conduc
## 1     1     Top  0-10 T0     1 5.40 0.188 0.92 215 16.35 7.65 0.72 1.14   1.09
## 2     1     Top  0-10 T0     2 5.65 0.165 1.04 208 12.25 5.15 0.71 0.94   1.35
## 3     1     Top  0-10 T0     3 5.14 0.260 0.95 300 13.02 5.68 0.68 0.60   1.41
## 4     1     Top  0-10 T0     4 5.14 0.169 1.10 248 11.92 7.88 1.09 1.01   1.64
## 5     2     Top 10-30 T1     1 5.14 0.164 1.12 174 14.17 8.12 0.70 2.17   1.85
## 6     2     Top 10-30 T1     2 5.10 0.094 1.22 129  8.55 6.92 0.81 2.67   3.18
```

First, here is a simple boxplot with default colors. I will change the colors using the resources shown below.

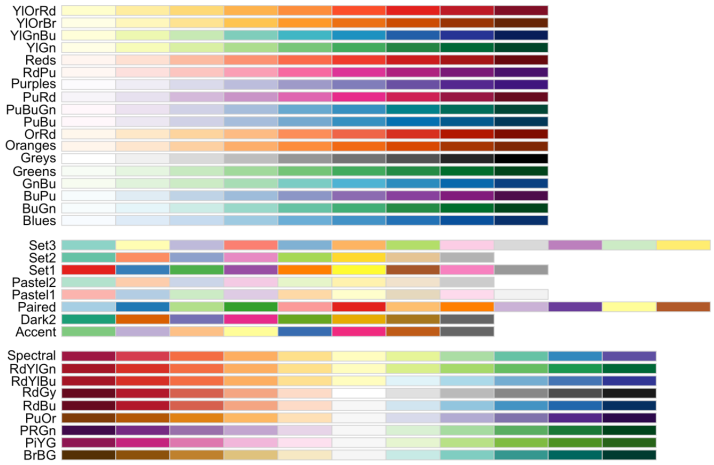
```
boxplot <-
  ggplot(Soils, aes(x = Depth, y = N, fill = Depth))+
  geom_boxplot()+
  ylab("N (%)")+
  xlab("Depth (cm)")


boxplot
```



Premade palettes EASIEST

RColorbrewer was one of the first packages to offer premade palettes. See some options below and here is the <documentation:https://www.rdocumentation.org/packages/RColorBrewer/versions/1.1-2/topics/RColorBrewer>. I don't use this resources I'll move on to demonstrating the resources I do use.

```
display.brewer.all()
```

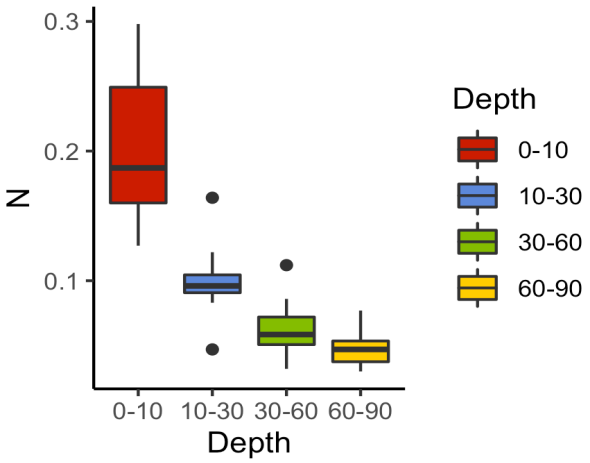


I prefer using the palettes in the ggsci package b/c the documentation is a little simpler/more modern.

Type vignette("ggsci") as a line of code to open documentation in R Studio. Note that you can also access graphical info for ggplot in the same way by running: vignette("ggplot2-specs")

This package "knows" to only provide as many colors as you have have categories. They aren't huge palettes so these won't work if you have a lot of groups. Note the ggsci package was loaded at the top of the script.

```
ggplot(Soils, aes(x = Depth, y = N, fill = Depth))+
  geom_boxplot()+
  theme_classic()+
  scale_fill_startrek()  # adding ggsci theme
```

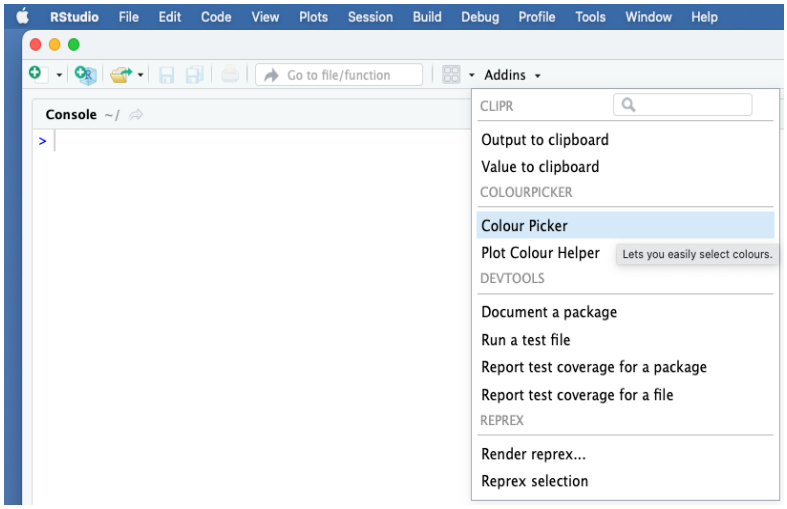


3 resources for finding & making custom palettes

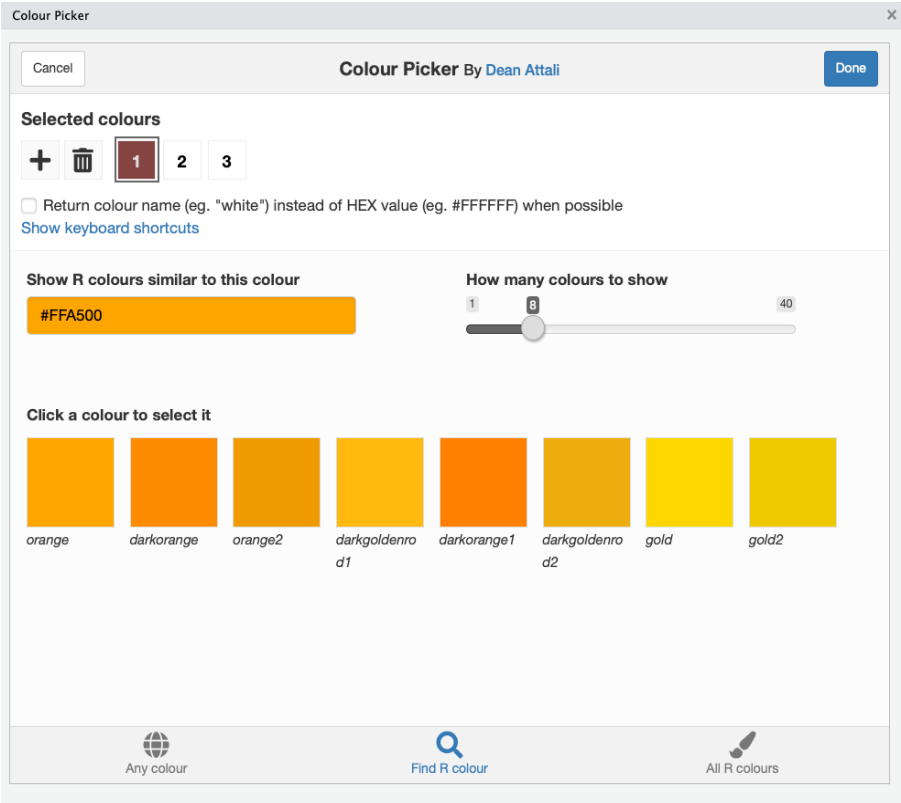
1. Colour Picker, a point-and-click addin for R Studio

EASIEST

Note: the colourpicker package was loaded at the top of the script. Once you install the colorpicker package you should have access to the 'Colour Picker' option in the the Addins menu at the top of your R Studio.



This will load a pop-out box that where you can build a discrete (categorical) palette using one of 3 options by clicking between the 3 tabs at the bottome of the box. The features are user friendly. When you have a palette you like, click 'Done' in the top right corner and it will automatically place the hex names of your color palette into a vector in the sript you you have open in R Studio. So handy!



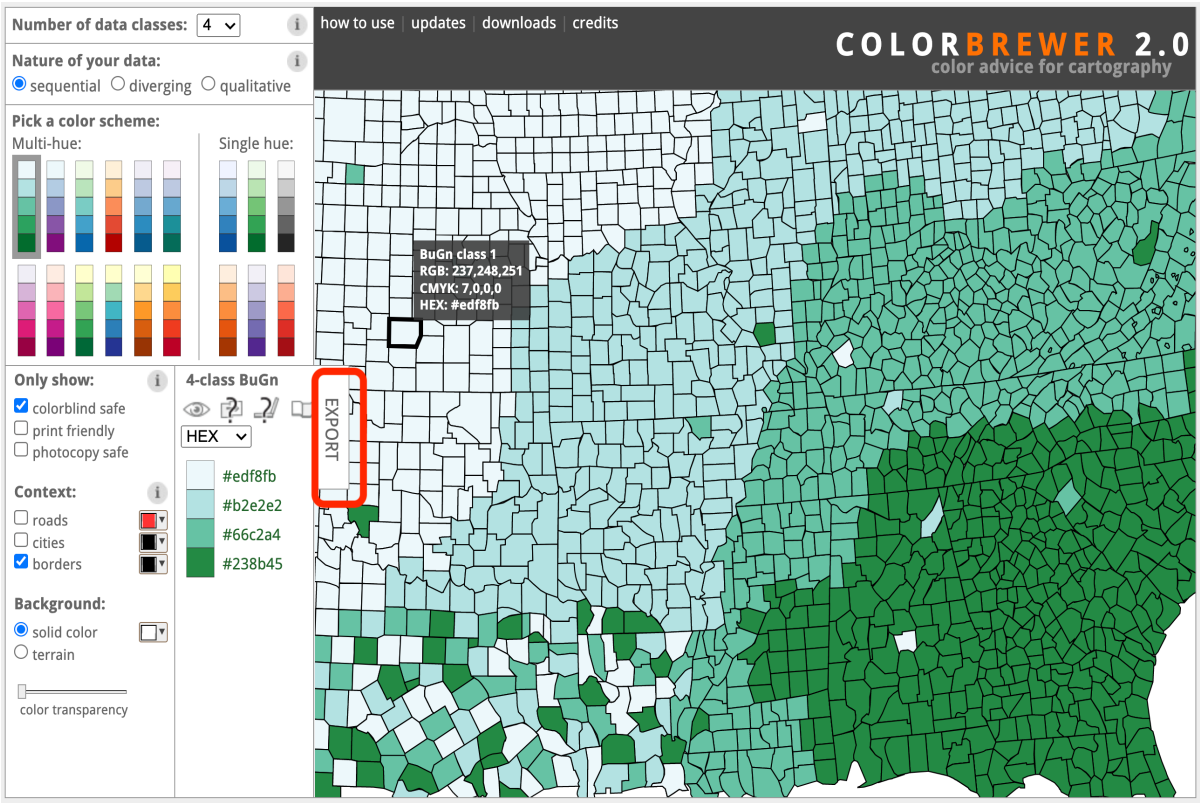
I then saved my 4 color palette to a name so I can add it to my graph later.

```
colorx4 <- c("#411085", "#46F0ED", "#FABC3F", "#E3FAC3")
```

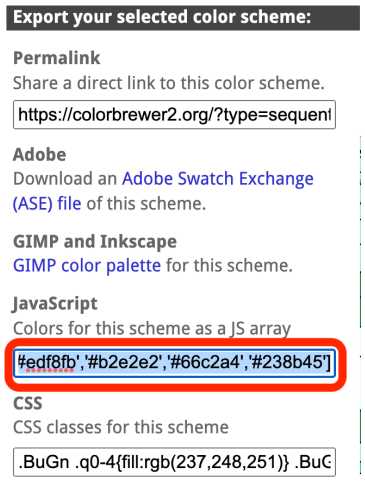
2. Colorbrewer2.org, a simple webpage INTERMEDIATE

https://colorbrewer2.org/

I like that you can easily select key aspects of a palette like the number of colors you want (up to 9) and it returns the highest contrasting values/hues for the number you select. It also has a colorblind safe criteria which is rare.

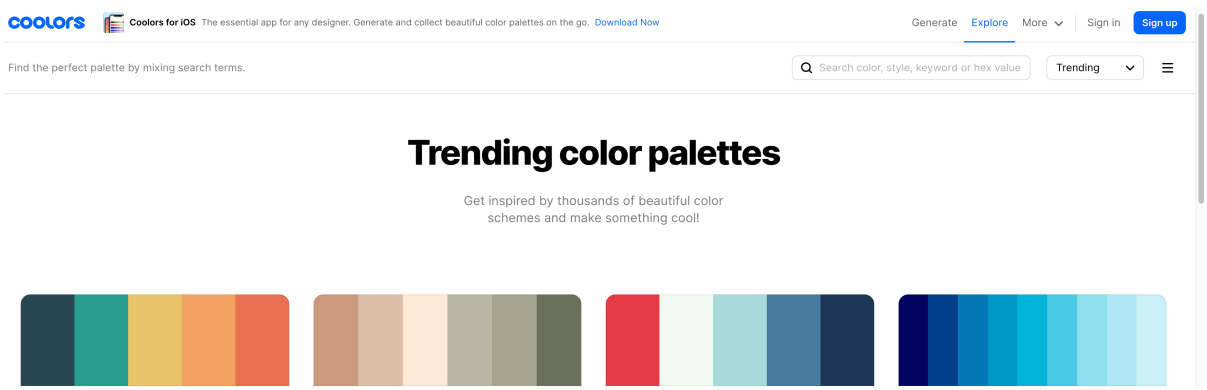


You can transfer the palette hex names fairly easily using the export feature (circled in red above). There are multiple options, but since I want to concatenate the names into a vector in my R script the easist option seems to be to copy/paste the javaScript formatted line and change the square brackets to parenthesis manually in my script.



3. Coolers, a comprehensive website ADVANCED

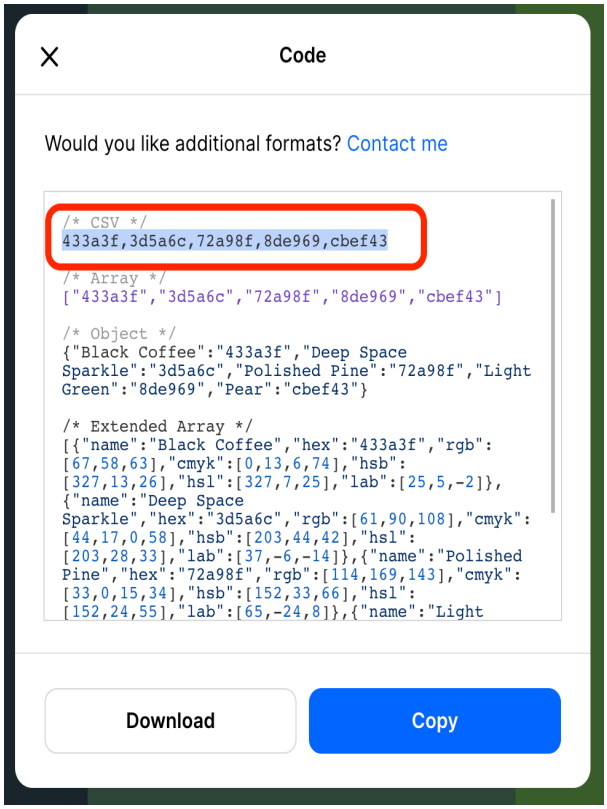
https://coolers.co



There's a lot of functionality provided on this website. Click on the explore menu to scroll through a ton of existing palettes. You can easily see the color names and fairly easily modify the palettes. Like colorbrewer2.org, you can select the colorblind friendly option (click on glasses icon under the Generate menu). Note that Deuteranomaly is the most common type of colorblindness which make red appear greenish.

The color picker option in the More menu allows you to see variation of a single color.

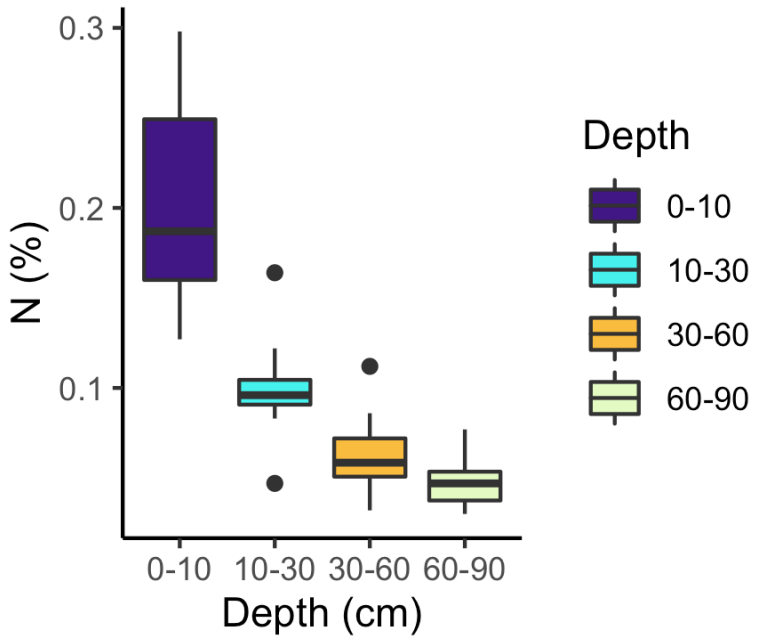
Regardless of the menu you are in, you should see an Export Icon on the page. Again, I would choose the export code option, and then copy/paste the comma separated hex names as shown below.



Apply your palette to a graph

Regardless of your approach to finding/making a color palette, here is an easy way you can apply it to your graphs.

```
boxplot+                                    # basic plot code saved in Setup
section
      scale_fill_manual(values = colorx4)+  # colorx4 was saved in Colourpicker
section above
      theme_classic() # adding my favorite theme to make it pretty
```



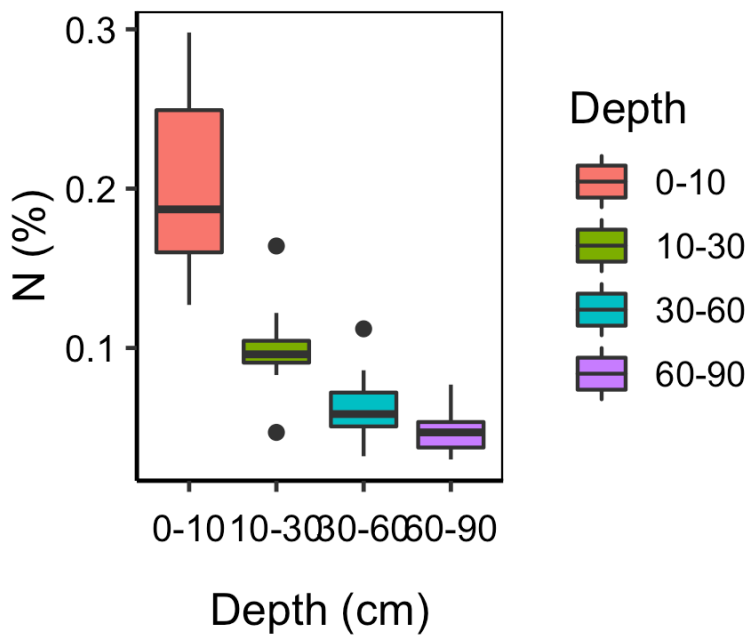
Make and save custom theme

This a nifty way to apply the same formatting to multiple graphs, say for all figures in a manuscript that you may wnt to have the same font, font size, plot background color, etc. It requires specifying everything you specify on the theme() argument line of ggplot2 code and putting this code in a homemade function. Honestly, it's a great way to practice building a function if you are new to functions because you can copy/paste the first 3 lines below and modify the rest (what you put inside the theme() argument).

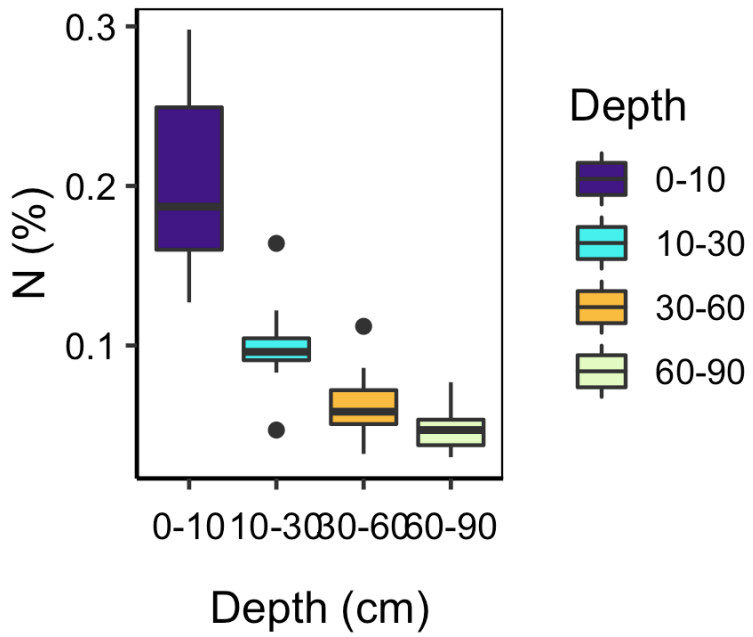
Save your theme to an object name like "theme_proj1" and then you can add your theme to any graph, just like you would add one of the provided themes like theme_bw() or my favorite, theme_classic().

In the example below you can see that I modified theme_classic() to make a full rectangle around the plot area instead of the default x and y axis lines only. I also made the axis font sizes larger because I find they are often a little small. Here are the resouces I used to learn how to make my theme: https://joeystanley.com/blog/custom-themes-in-ggplot2 https://rpubs.com/mclaire19/ggplot2-custom-themes

```
theme_proj1 <- function(base_size = 12, base_family = ""){
  font = "Arial"
  theme_classic(base_size=12, base_family="") %+replace%
#  theme_classic()+                        # clean, modern plot theme (not gray
background now)

  theme(
    panel.border = element_rect(fill = NA, color = "black"),

      axis.text = element_text(              #axis text
                  family = font,             #axis famuly
                  size = 10),                 #font size

      axis.text.x = element_text(            #margin for axis text
                   margin=margin(5, b = 10)))
}

boxplot +
  theme_proj1()
```




```
boxplot +
  theme_proj1()+
  scale_fill_manual(name = "Depth",
                    values = colorx4)
```



More info on colorblind visualizations

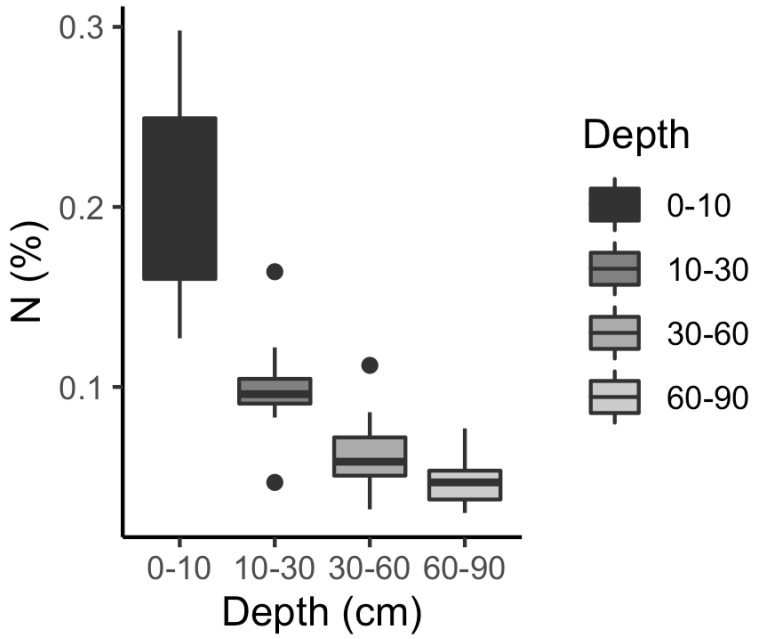
https://towardsdatascience.com/two-simple-steps-to-create-colorblind-friendly-data-visualizations-2ed781a167ec

Greyscale graphs

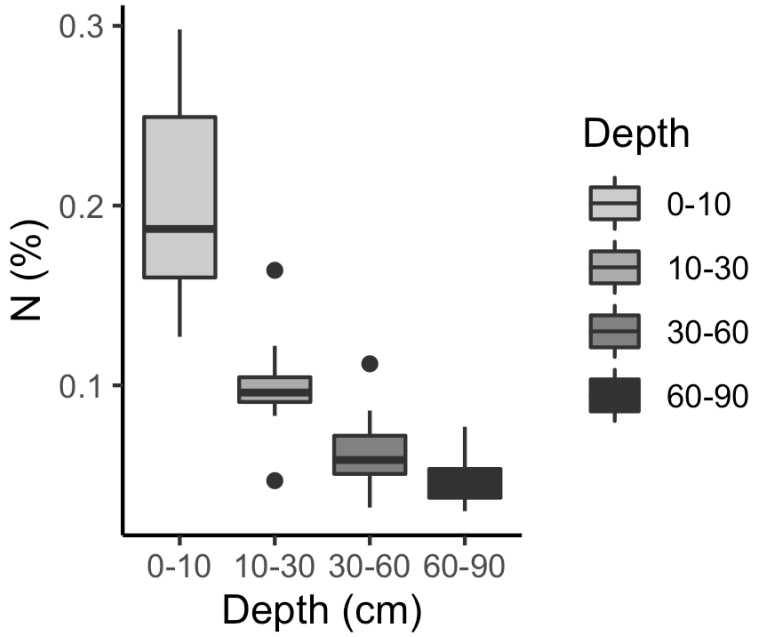
If you are publishing in a journal that requires you to pay for color images (for the tiny percentage of people who still get printed versions of the journal issues), you may choose to remake some of your figures without color. Here are some quick ways to do this.

Use scale_fill_grey() to fill bars and boxplots and scale_color_grey for points. Both options are for discrete scales (aka categorical variables).

```
# Boxplot

boxplot+
  theme_classic()+
  scale_fill_grey()
```



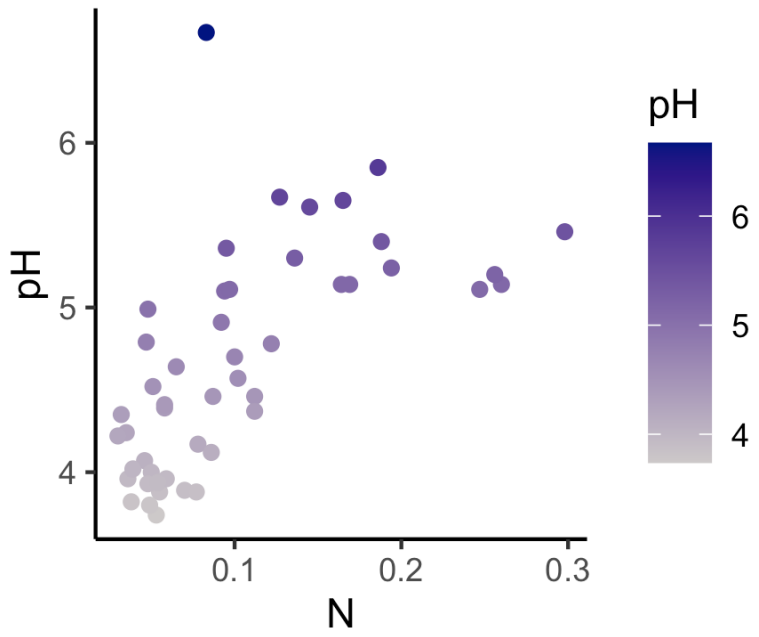

```
# Boxplot with reversed gradient
boxplot+
  theme_classic()+
  scale_fill_grey(start = 0.8, end = 0.2)
```



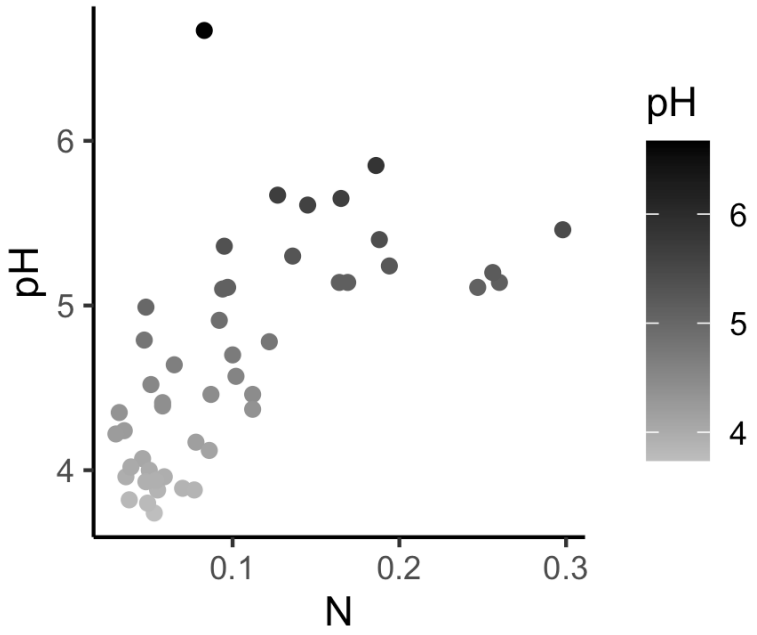
Applying continuous scale palettes

Since I didn't address this anywhere previously, I'll share here that the easiest way I've found to modify continuous color palettes (a gradient as opposed to categories) is to specify a starting color and end ending color in ggplot's scale_colour_gradient() argument. This also allows you to easily control which end of color gradient you want to be dark value vs. light.

```
# color gradient
ggplot(Soils, aes(x = N, y = pH, col = pH))+
  theme_classic()+
  geom_point()+
scale_colour_gradient(low = "snow3", high = "Navy")
```




```
# make a grayscale
ggplot(Soils, aes(x = N, y = pH, col = pH))+
  theme_classic()+
  geom_point()+
  scale_color_gradient(low = "grey", high = "black")
```


pH
6
5
4
0.1
0.2
0.3
N
pH
6
5
4