

# R color cheatsheet

Finding a good color scheme for presenting data can be challenging. This color cheatsheet will help!

## R uses hexadecimal to represent colors

Hexadecimal is a base-16 number system used to describe color. Red, green, and blue are each represented by two characters (#rrggbb). Each character has 16 possible symbols: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F:

"00" can be interpreted as 0.0 and "FF" as 1.0  
i.e., red = #FF0000, black = #000000, white = #FFFFFF

Two additional characters (with the same scale) can be added to the end to describe transparency (#rrggbbaa)

## R has 657 built in color names

To see a list of names:

`colors()`

These colors are displayed on P. 3.

Example:

peachpuff4

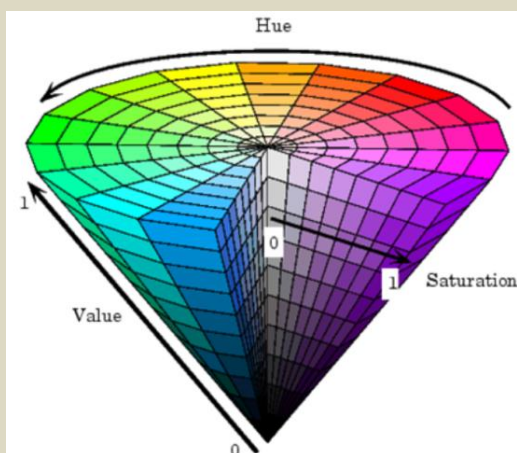
## R translates various color models to hex, e.g.:

- RGB (red, green, blue): The default intensity scale in R ranges from 0-1; but another commonly used scale is 0-255. This is obtained in R using `maxColorValue=255`. *alpha* is an optional argument for transparency, with the same intensity scale.  
`rgb(r, g, b, maxColorValue=255, alpha=255)`
- HSV (hue, saturation, value): values range from 0-1, with optional alpha argument  
`hsv(h, s, v, alpha)`
- HCL (hue, chroma, luminance): hue describes the color and ranges from 0-360; 0 = red, 120 = green, blue = 240, etc. Range of chroma and luminance depend on hue and each other  
`hcl(h, c, l, alpha)`

### A few notes on HSV/HLC

HSV is a better model for how humans perceive color. HCL can be thought of as a perceptually based version of the HSV model....blah blah blah...

Without delving into color theory: color schemes based on HSV/HLC models generally just look good.



R can translate colors to rgb (this is handy for matching colors in other programs)

`col2rgb(c("#FF0000", "blue"))`

## R Color Palettes

This is for all of you who don't know anything about color theory, and don't care but want some nice colors on your map or figure....NOW!

**TIP:** When it comes to selecting a color palette, **DO NOT** try to handpick individual colors! You will waste a lot of time and the result will probably not be all that great. R has some good packages for color palettes. Here are some of the options

### Packages: grDevices and colorRamps

grDevices comes with the base installation and colorRamps must be installed. Each palette's function has an argument for the number of colors and transparency (*alpha*):

`heat.colors(4, alpha=1)`  
> #FF0000FF" "#FF8000FF" "#FFFF00FF" "#FFFF80FF"

grDevices  
palettes  
cm.colors  
topo.colors  
terrain.colors  
heat.colors  
rainbow  
see P. 4 for  
options

For the `rainbow` palette you can also select start/end color (red = 0, yellow = 1/6, green = 2/6, cyan = 3/6, blue = 4/6 and magenta = 5/6) and saturation (s) and value (v):  
`rainbow(n, s = 1, v = 1, start = 0, end = max(1, n - 1)/n, alpha = 1)`

### Package: RcolorBrewer

This function has an argument for the number of colors and the color palette (see P. 4 for options).  
`brewer.pal(4, "Set3")`

> "#8DD3C7" "#FFFFB3" "#BEBADA" "#FB8072"

To view colorbrewer palettes in R: `display.brewer.all(5)`

There is also a very nice interactive viewer:

<http://colorbrewer2.org/>

## ## My Recommendation ##

### Package: colorspace

These color palettes are based on HCL and HSV color models. The results can be very aesthetically pleasing. There are some default palettes:

`rainbow_hcl(4)`  
"#E495A5" "#ABB065" "#39BEB1" "#ACA4E2"

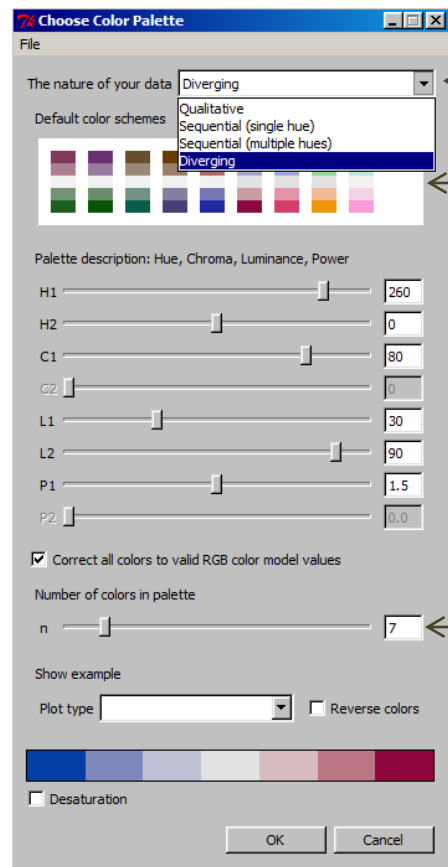
colorspace  
default palettes  
diverge\_hcl  
diverge\_hsl  
terrain\_hcl  
sequential\_hcl  
rainbow\_hcl

However, all palettes are fully customizable:  
`diverge_hcl(7, h = c(246, 40), c = 96, l = c(65, 90))`  
Choosing the values *would* be daunting. But there are some recommended palettes in the colorspace documentation. There is also an interactive tool that can be used to obtain a customized palette. To start the tool:  
`pal <- choose_palette()`

# R color cheatsheet

## Overview of colorspace palette selector

```
library("colorspace")
pal <- choose_palette()
```

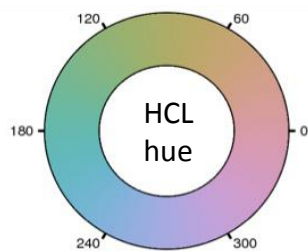


Select the type of color scheme based on the type of data

Default color schemes – can be used “as is” or as a starting point for modification

Interactively select:

- hue: color
- chroma: low chroma = gray
- luminance: high luminance = pastel
- power: how the color changes along a gradient

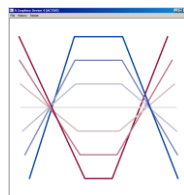


Select # of colors in palette

- Save palette for future R sessions:
- txt file with hex codes
  - .R file with a function describing how to generate the palette.
- `source` can be used to import the function into R; but one complication is that you have to open the .R file and name the function to use it.

- Copy values into relevant colorspace functions.
- Diverging color schemes:  
`diverge_hcl(7, h = c(260, 0), c = 100, l = c(28, 90), power = 1.5)`  
 Sequential color schemes:  
`sequential_hcl(n, h, c = c(), l = c(), power)`  
 Qualitative color schemes:  
`rainbow_hcl(n, c, l, start, end)` (for qualitative schemes; start/ end refer to the H1/H2 hue values)

Display color scheme with different plot types



When “OK” is selected, the color palette will be saved in the R session. To return 7 hex color codes from the selected palette:

```
pal <- choose_palette()
pal(7)
```

[NOTE: These values are not saved if you don’t save the session]

## How to use hex codes to define color using the plot function

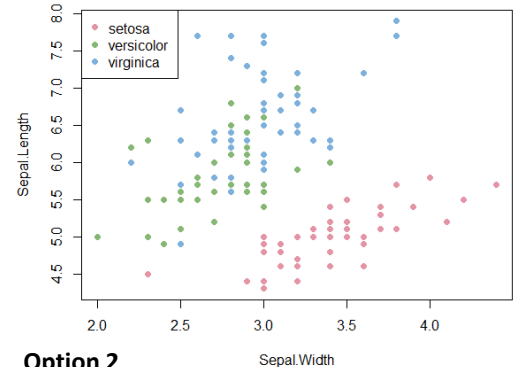
### Discrete variables

#### Option 1

If you don’t need to control which colors are associated with each level of a variable:

```
plot(Sepal.Length ~ Sepal.Width,
     col=rainbow_hcl(3)[c(Species)],
     data=iris, pch=16)
```

```
legend("topleft", pch=16, col=rainbow_hcl(3),
      legend=unique(iris$Species))
```



#### Option 2

If you want to control which colors are associated with the levels of a variable, I find it easiest to create a variable in the data:

```
iris$color <- factor(iris$Species,
                    levels=c("virginica", "versicolor", "setosa"),
                    labels=rainbow_hcl(3))
```

```
plot(Sepal.Length ~ Sepal.Width,
     col=as.character(color), pch=16, data=iris)
```

### Continuous variables

#### Option 1

Break into categories and assign colors:

```
iris2 <- subset(iris, Species=="setosa")
```

```
color <- cut(iris2$Petal.Length,
            breaks=c(0,1.3,1.5,2), labels=sequential_hcl(3))
```

Or, break by quantiles (be sure to include 0 & 1):

```
color <- cut(iris2$Petal.Length,
            breaks=quantile(iris$Petal.Length, c(0, 0.25, 0.5, 0.75, 1)), labels=sequential_hcl(3))
```

```
plot(Sepal.Width ~ Sepal.Length, pch=16,
     col=color, data=iris2)
```

#### Option 2

Fully continuous gradient:

```
data <- data.frame("a"=runif(10000),
                  "b"=runif(10000))
```

```
color=diverge_hcl(length(data$a))[rank(data$a)]
plot(a~b, col=color, pch=16, data=data)
```

For ggplot2, I think the most flexible color scales are:

```
scale_colour_manual
scale_colour_gradient
```

for discrete and continuous variables, respectively



# grDevices::colors

coral3	gray27	gray39	lightpink1	mistyrose1	pink4	slategray1	yellowgreen
coral2	gray26	gray38	lightpink	mistyrose	pink3	slategray	yellow4
coral1	gray25	gray37	lightgray	mintcream	pink2	slateblue4	yellow3
coral	gray24	gray36	lightgreen	pink1	pink	slateblue3	yellow2
chocolate4	gray23	gray35	lightgray	mediumvioletred	peachpuff4	slateblue2	yellow1
chocolate3	gray22	gray34	lightgoldenrod4	mediumspringgreen	peachpuff3	slateblue1	whitesmoke
chocolate2	gray21	gray33	lightgoldenrod3	mediumturquoise	peachpuff2	skyblue4	wheat4
chocolate1	gray20	gray32	lightgoldenrod2	mediumslateblue	peachpuff1	skyblue3	wheat3
chocolate	gray19	gray31	lightgoldenrod1	mediumseagreen	peachpuff	skyblue2	wheat2
chartreuse4	gray18	gray30	lightgoldenrod	mediumpurple4	papayawhip	skyblue1	wheat1
chartreuse3	gray17	gray29	lightcyan4	mediumpurple1	palevioletred4	sienna4	violetred4
chartreuse2	gray16	gray28	lightcyan3	mediumpurple2	palevioletred3	sienna3	violetred3
chartreuse1	gray15	gray27	lightcyan2	mediumpurple3	palevioletred2	sienna2	violetred2
chartreuse	gray14	gray26	lightcyan1	mediumpurple4	palevioletred1	sienna1	violetred1
cadetblue4	gray13	gray25	lightcyan	mediumorchid4	palevioletred	sienna	violet
cadetblue3	gray12	gray24	lightcoral	mediumorchid1	paleturquoise4	seashell4	turquoise4
cadetblue2	gray11	gray23	lightblue4	mediumorchid2	paleturquoise3	seashell3	turquoise3
cadetblue1	gray10	gray22	lightblue3	mediumorchid3	paleturquoise2	seashell2	turquoise2
burlywood4	gray9	gray21	lightblue2	mediumorchid	paleturquoise1	seashell1	turquoise1
burlywood3	gray8	gray20	lightblue1	mediumblue	paleturquoise	seashell	tomato4
burlywood2	gray7	gray19	lightblue	maroon4	paleturquoise	sandybrown	tomato3
burlywood1	gray6	gray18	lemonchiffon4	maroon3	palegreen4	salmon4	tomato2
burlywood	gray5	gray17	lemonchiffon3	maroon2	palegreen3	salmon3	tomato1
brown4	gray4	gray16	lemonchiffon2	maroon1	palegreen2	salmon2	tomato
brown3	gray3	gray15	lemonchiffon1	maroon	palegreen1	salmon1	thistle4
brown2	gray2	gray14	lemonchiffon	magenta4	palegreen	salmon	thistle3
brown1	gray1	gray13	lawngreen	magenta3	palegreen	thistle2	thistle1
brown	gray0	gray12	lavenderblush4	magenta2	orchid4	thistle	tan4
blueviolet	gray	gray11	lavenderblush3	magenta1	orchid3	tan3	tan2
blue4	goldenrod4	gray10	lavenderblush2	linen	orchid2	tan1	tan
blue3	goldenrod3	gray9	lavenderblush1	limegreen	orchid1	steelblue4	steelblue3
blue2	goldenrod2	gray8	lavender	lightyellow4	orange4	steelblue2	steelblue1
blue1	goldenrod1	gray7	khaki3	lightyellow3	orange3	steelblue	springgreen4
blue	goldenrod	gray6	khaki2	lightyellow2	orange2	springgreen3	springgreen2
blanchedalmond	gold4	gray5	khaki1	lightyellow1	orange1	springgreen1	springgreen
black	gold3	gray4	khaki	lightsteelblue4	orange	snow4	snow3
bisque4	gold2	gray3	ivory4	lightsteelblue3	orange3	snow3	snow2
bisque3	gold1	gray2	ivory3	lightsteelblue2	orange2	snow2	snow1
bisque2	ghostwhite	gray1	ivory2	lightsteelblue1	orange1	snow	slategray4
bisque1	gainsboro	gray0	ivory1	lightslategray	orange	slategray3	slategray2
bisque	forestgreen	greenyellow	ivory	lightslategray	orange	slategray	
beige	floralwhite	green4	indianred4	lightslategray	orange		
azure4	firebrick4	green3	indianred3	lightslategray	orange		
azure3	firebrick3	green2	indianred2	lightslategray	orange		
azure2	firebrick2	green1	indianred1	lightslategray	orange		
azure1	firebrick1	green	indianred	lightslategray	orange		
azure	firebrick	gray100	indianred	lightslategray	orange		
aquamarine4	dodgerblue4	gray99	hotpink4	lightslategray	orange		
aquamarine3	dodgerblue3	gray98	hotpink3	lightslategray	orange		
aquamarine2	dodgerblue2	gray97	hotpink2	lightslategray	orange		
aquamarine1	dodgerblue1	gray96	hotpink1	lightslategray	orange		
aquamarine	dodgerblue	gray95	hotpink	lightslategray	orange		
antiquewhite4	cornsilk4	gray94	honeydew4	lightslategray	orange		
antiquewhite3	cornsilk3	gray93	honeydew3	lightslategray	orange		
antiquewhite2	cornsilk2	gray92	honeydew2	lightslategray	orange		
antiquewhite1	cornsilk1	gray91	honeydew1	lightslategray	orange		
antiquewhite	cornsilk	gray90	honeydew	lightslategray	orange		
aliceblue	cornflowerblue	gray89	gray100	lightslategray	orange		
white	coral4	gray88		lightslategray	orange		

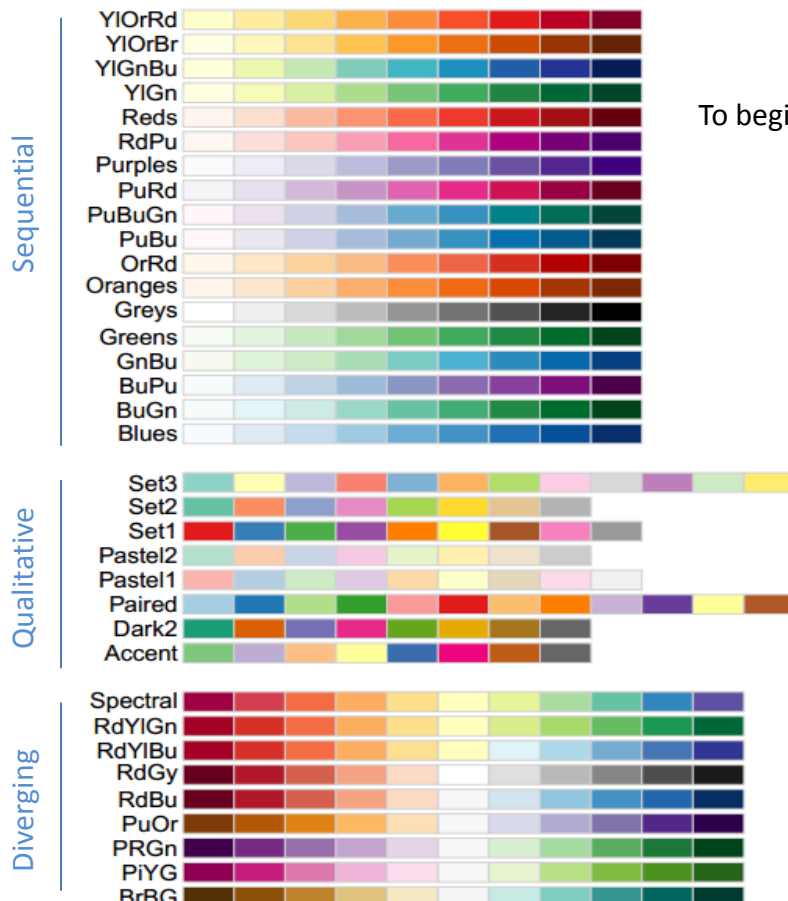


## colorRamps and grDevices



colorRamps and grDevices color palette, display from:  
<http://bc.bojanorama.pl/2013/04/r-color-reference-sheet/>

## RColorBrewer



To display RColorBrewer palette: `display.brewer.all()`  
 For interactive color selector: <http://colorbrewer2.org/>

## colorspace defaults



## colorspace useful palette examples



To begin interactive color selector: `pal <- choose_palette()`

## Useful Resources:

A larger color chart of R named colors:

<http://research.stowers-institute.org/efg/R/Color/Chart/ColorChart.pdf>

Nice overview of color in R:

<http://research.stowers-institute.org/efg/Report/UsingColorInR.pdf>

[http://students.washington.edu/mclarkso/documents/colors\\_Ver2.pdf](http://students.washington.edu/mclarkso/documents/colors_Ver2.pdf)

A color theory reference:

Zeileis, A. K. Hornik, P. Murrell. 2009. Escaping RGBland: selecting colors for statistical graphics. Computational and Statistics & Data Analysis 53:3259-3270