Color in R

Jonathan M. Lees
University of North Carolina, Chapel Hill
Department of Geological Sciences
CB #3315, Mitchell Hall
Chapel Hill, NC 27599-3315
email: jonathan.lees@unc.edu
ph: (919) 962-0695

May 17, 2011

1 Introduction

Color can be completely controled on nearly all illustrations or graphics.

Care should be taken so that information conveyed is properly illustrated with the appropriate graphics.

Color may appear different in print than on a projected figure in a dark room. On a projected presentation, colors may also vary from projector to projector, depending on settings in the projector, strength of the light source and other parameters not under the control of the author.

Sometimes it is useful to have flexible color schemes. If journals charge high fees for color graphics, be prepared to change to black/white and grey scale figures.

2 Color definitions

There are several ways to control color in R.

2.1 Default Palette

```
> options(width=60)
> options(continue=" ")
> options(SweaveHooks=list(fig=function()
 par(mar=c(5.1, 4.1, 1.1, 2.1))))
> require(graphics)
> library(RPMG)
```

The default palette, set when you start R with no other changes is defined as:

```
> palette()
[1] "black"   "red"     "green3"  "blue"    "cyan"
[6] "magenta" "yellow"  "gray"
```

2.2 color by number

Once a default palette is defined, colors can be accessed by refering to the index of the color in the default palette. Normally object colors are set by defining col in the call to plot. here we add another parameter bg (background) to color the inside of the points.

```
> set.seed(3)
> x = 1:10
> y = runif(length(x))
> ## plot(x, y, col=1:10, pch=21, cex=2)
> plot(x, y, bg=1:10, pch=21, cex=2)
```

The colors in the default palette are very basic. The can produce plots that do not look very good.

Colors in R are cycled through the default palette when they are called by number. In this case there are eight colors and when there are 10 different calls the colors revert to the beginning and start over.

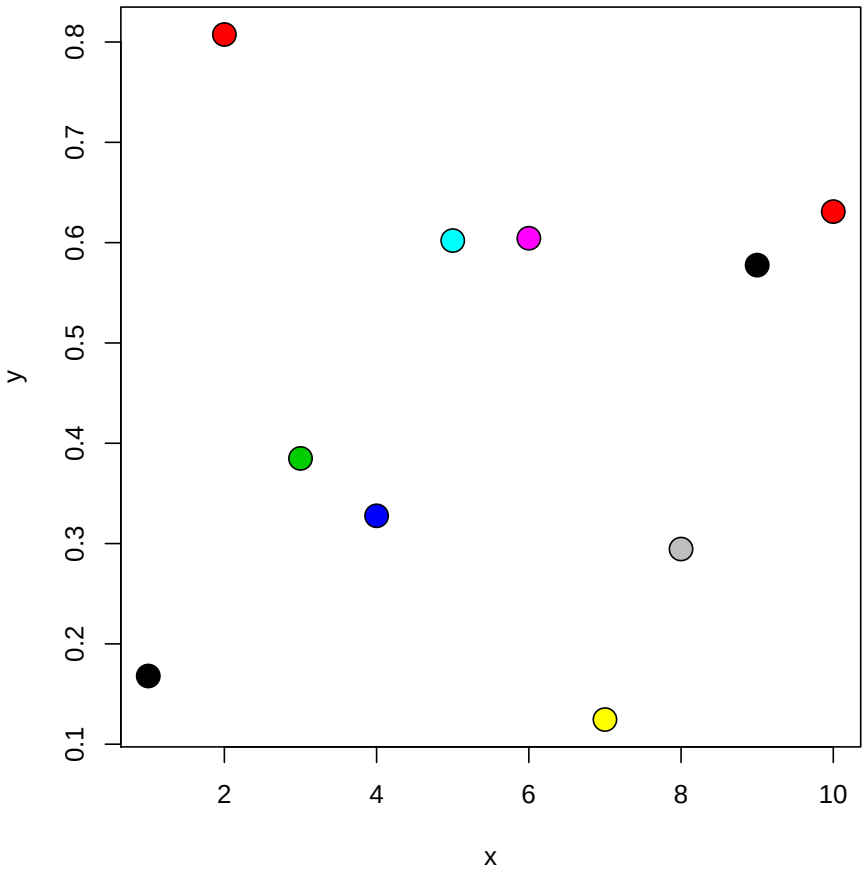


Figure 1: Default palette

2.3 color by name

Another way to call colors is by name, if they are defined in the system. Here I color the background color blue and let the border vary according to the default palette.

```
> mycols <- adjustcolor(palette(), alpha.f = 0.3)
> plot(runif(1000, 0, 10), runif(1000, min(y), max(y)), pch=21, cex=2, bg=grey
> points(x, y, bg=mycols[1:10],  pch=21, cex=2)
```

2.4 system colors

There are 657 named colors in the standard distribution of R and these can be accessed by a call to the function colors(). Since this is a large number of text strings, I will only list the first ten here:

```
> c1 = colors()
> print(c1[1:10])
 [1] "white"         "aliceblue"     "antiquewhite"
 [4] "antiquewhite1" "antiquewhite2" "antiquewhite3"
 [7] "antiquewhite4" "aquamarine"    "aquamarine1"
[10] "aquamarine2"
```

2.5 Defining colors

There are several systems available in R for defining colors. The standard R-G-B (rgb) system for computer screens is the one I normally use, although hsv, hcl and grey are also used for a variety of applications where details of color specifications are important. There are many web sites on line where one can read about these choices. See below.

RGB is chosen as fractions or percentages of red-green-blue, mised together. Additionally, the choice of "transparent" can be used to make a color transparent, obviously.

Example:

■fig=TRUE, echo=TRUE, results=verbatim■= c1 = rgb(0.1, 0.4, 0.7) c2 = rgb(0.4, 0.4, 0.1) plot(0, 0, type='n') rect(-.4, -.4, .4, .4, col=c1) rect(-.2, -.2, .2, .2, col=c2)

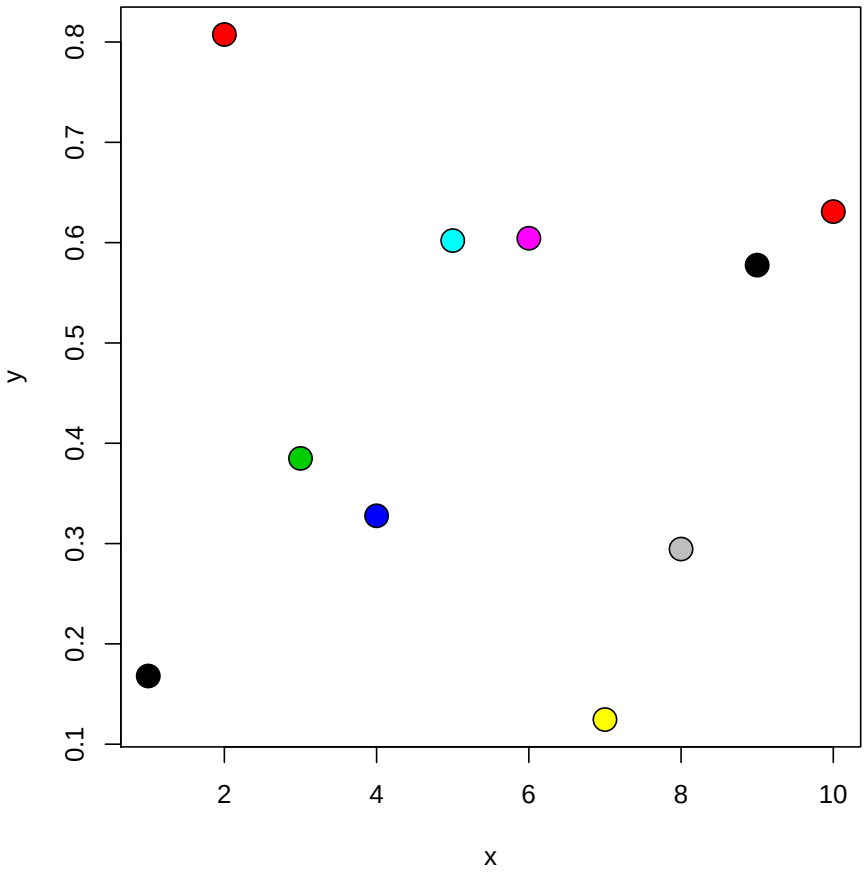
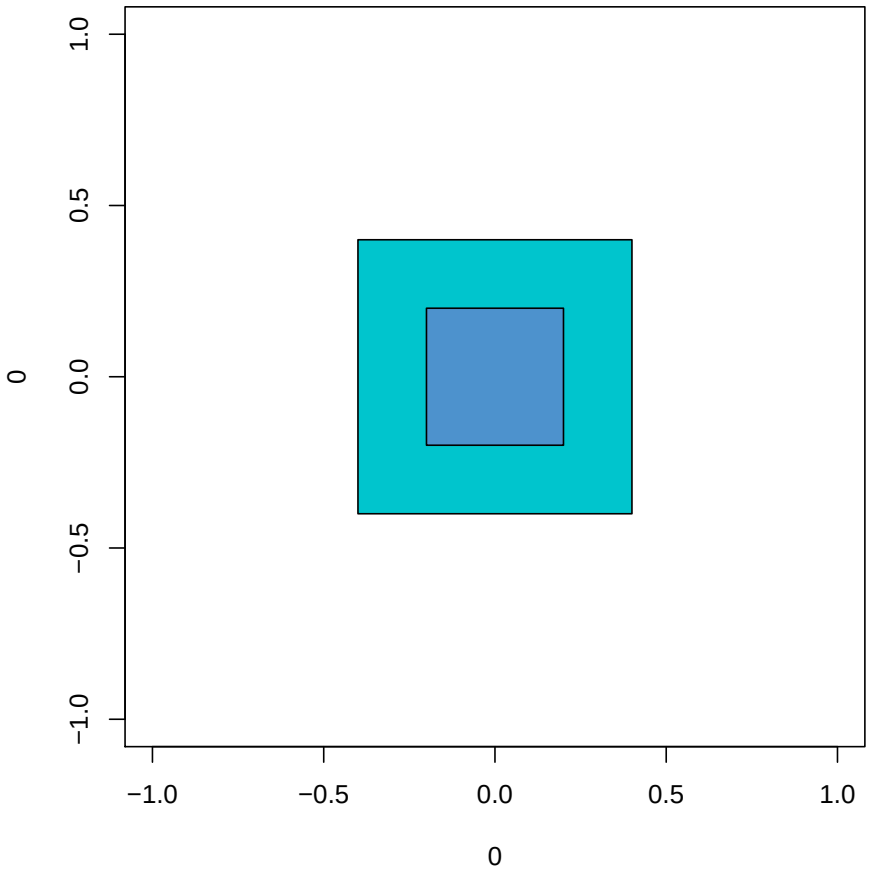


Figure 2: Default palette

In some situations it might be useful to modify colors. The function col2rgb converts named colors to a vector, ranging from 0 to 255. To convert these again, divide by 255 and use the rgb function.

```
> A1 = col2rgb("turquoise3")/255
> c1 = rgb(A1[1], A1[2], A1[3])
>  c2 = rgb(A1[1]+.3, A1[2]-.2, A1[3])
>  plot(0, 0, type='n')
>  rect(-.4, -.4, .4, .4, col=c1)
>  rect(-.2, -.2, .2, .2, col=c2)
```



2.6 Color palettes

It is always possible to change the palette by choosing a different set of named colors to serve as the defaults in subsequent plots or functions. Once the palette is defined and set it will be used from then on within that R session.

For example, here we set the colors to a specific palette make a plot and then revert to the original default palette.

```
> pdark =c( "black","darkmagenta","forestgreen","blueviolet","tan3",
   "lightseagreen","deeppink","cyan3","bisque3","magenta1",
   "lightsalmon3","darkcyan","darkslateblue","chocolate4",
   "goldenrod4","mediumseagreen" )
> palette(pdark)
> plot(x, y, bg=1:10,  pch=21, cex=2, lwd=1, col="transparent" )
> palette("default")
```

The user can control all features of the plotting region by changing the default settings in par.

```
> opar = par(no.readonly=TRUE)
> ###  for dark background:
> par(bg=rgb(0.1,0.1,0.1), fg=rgb(1,1,1),col.axis=rgb(1,1,1),col.lab=rgb(1,1,1
> plite =c( "grey","lightblue1","pink","darkseagreen2","gold1","chartreuse1","
> palette(plite)
> plot(x, y, bg=mycols[1:10],  pch=21, cex=5)
> par(opar)
```

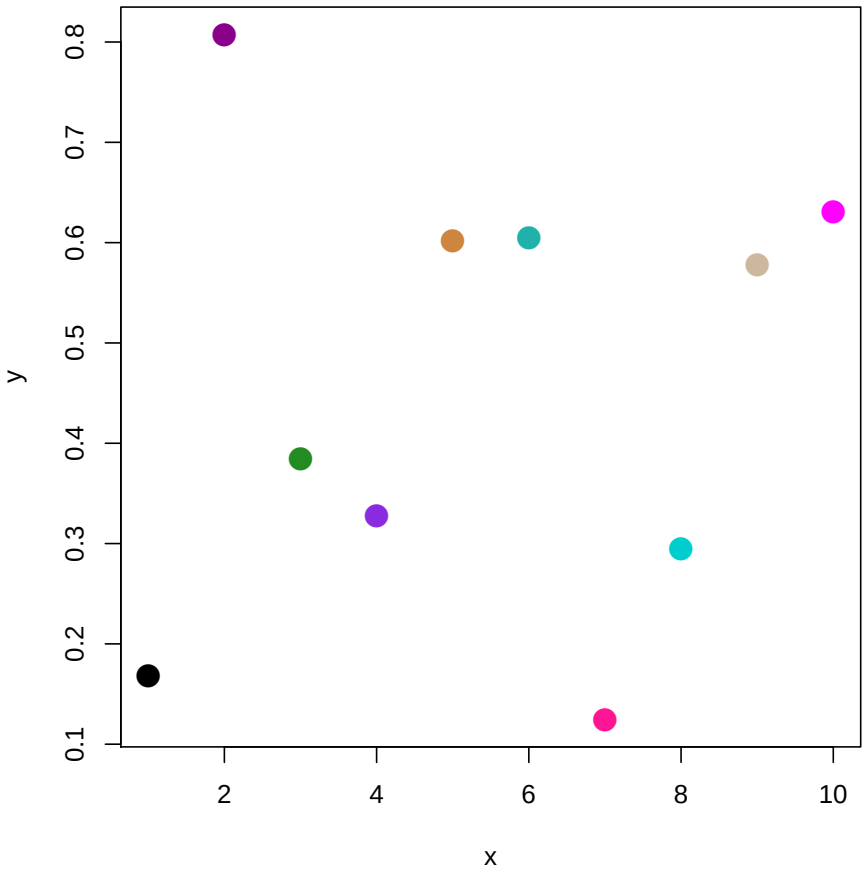
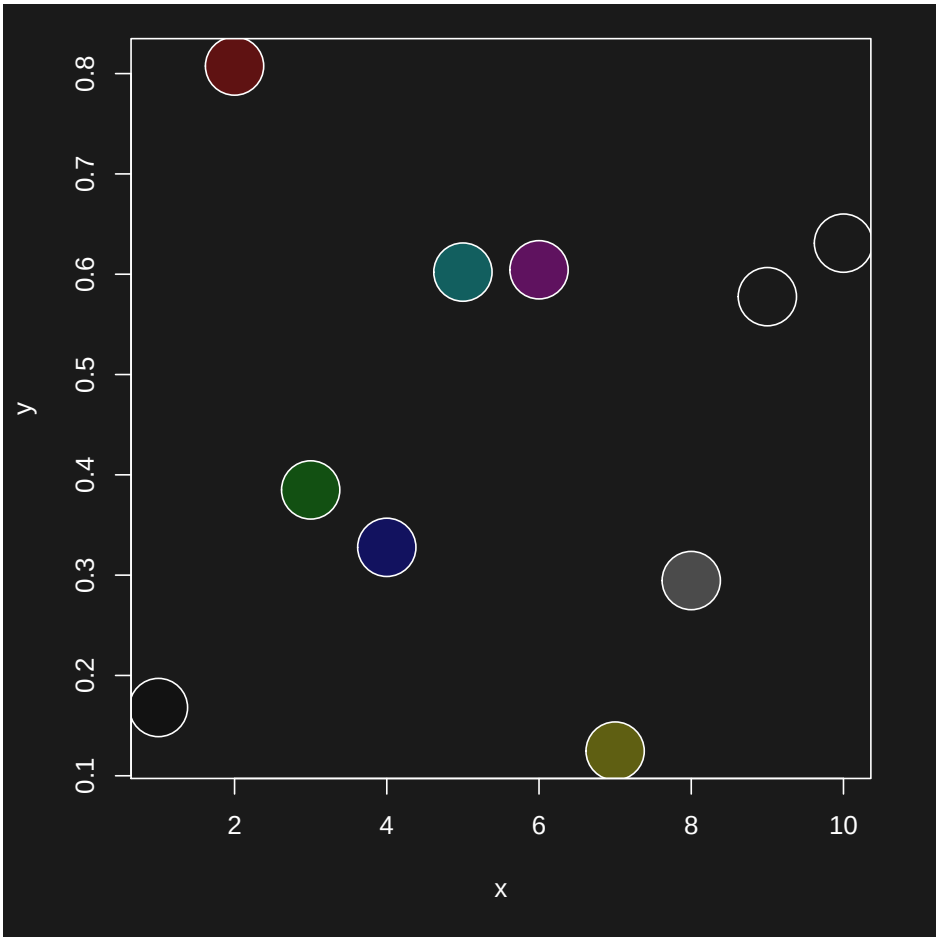
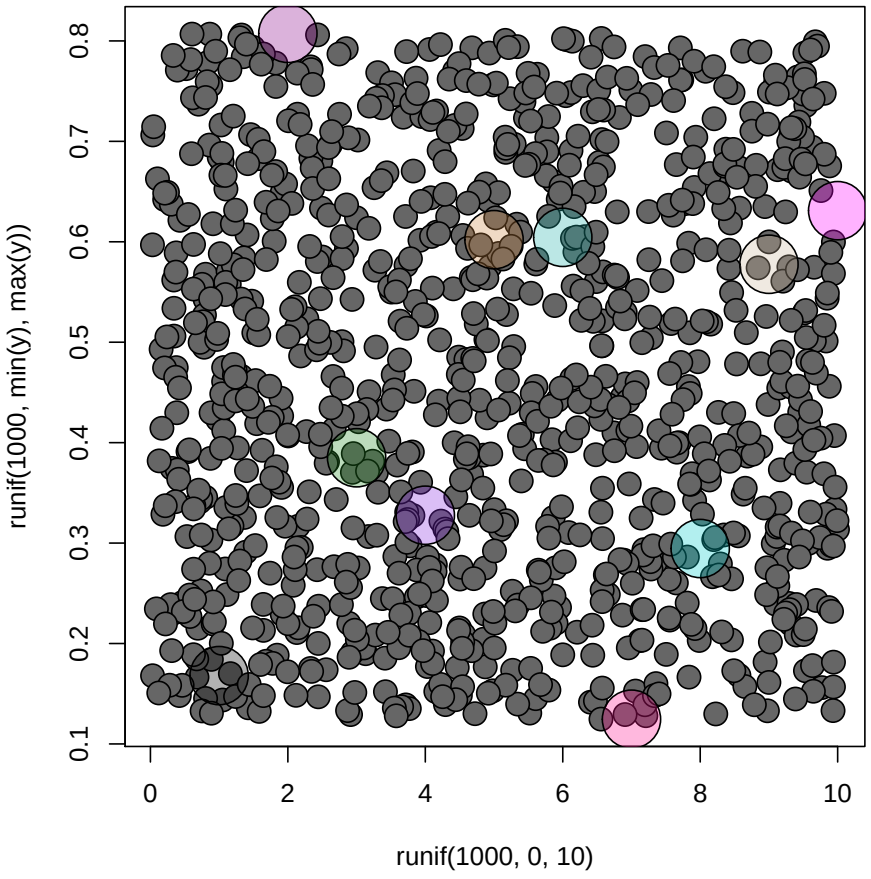


Figure 3: pdark palette



Transparency:

```
> palette(pdark)
> ######  transparency effect
> mycols <- adjustcolor(palette(), alpha.f = 0.3)
> plot(runif(1000, 0, 10), runif(1000, min(y), max(y)), pch=21, cex=2, bg=grey
> points(x, y, bg=mycols[1:10],  pch=21, cex=5)
```



2.7 creative palettes

You can create palettes for different uses depending on the work you are doing and the information you are trying to convey.

```
> SYSCOL = hcl.colors(100)
> SHOWPAL(SYSCOL, ncol=10)
```

```
> data(volcano)
> vx = 10*c(1:dim(volcano)[1])
> vy = 10*c(1:dim(volcano)[2])
> image(x=vx, y=vy, z=volcano, col=SYSCOL,  ann=FALSE, axes=FALSE, asp=1)
> contour(x=vx, y=vy, z=volcano, add=TRUE)
```

```
$N
[1] 100

$ncol
[1] 10

$nrow
[1] 10

$dx
[1] 0.1

$dy
[1] 0.1
```


Figure 4: hcl palette

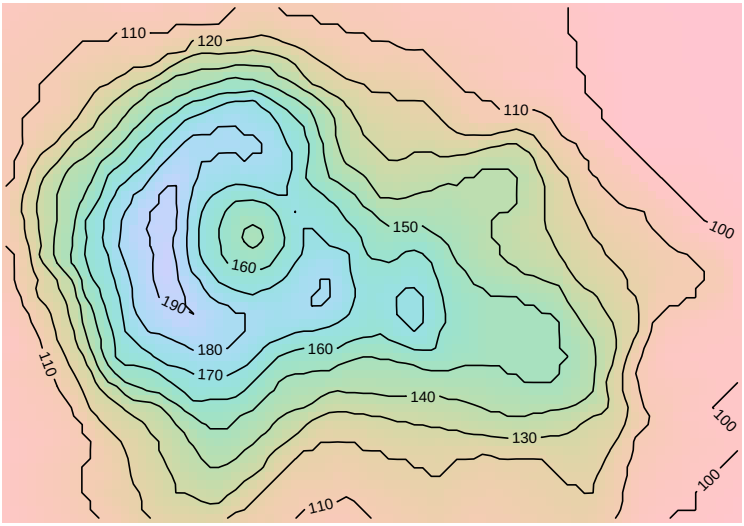


Figure 5: Volcano with hcl palette

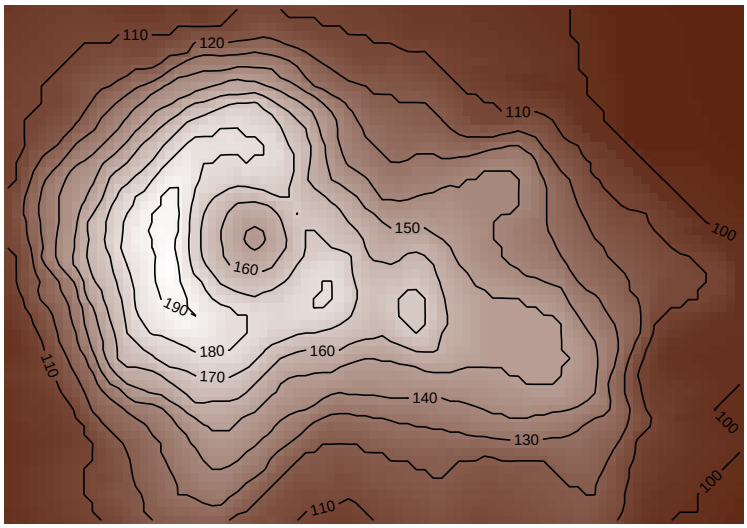


Figure 6: Volcano with sepia palette

```
> SYSCOL = sepia.colors(100)
> image(x=vx, y=vy, z=volcano, col=SYSCOL,  ann=FALSE, axes=FALSE, asp=1)
> contour(x=vx, y=vy, z=volcano, add=TRUE)
```

3 Links to other advice

Chart

HCL-Based Color Palettes in R