R语言中的色彩和调色板

Publish: February 23, 2014

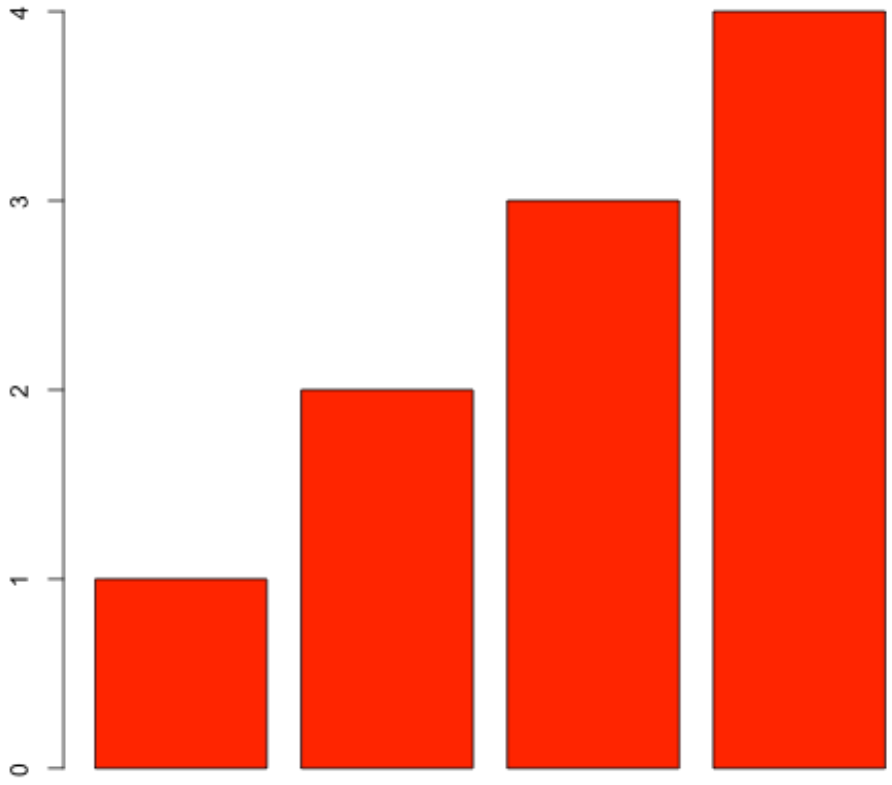
可视化数据时，色彩往往是最欠考虑的因素。的确，在一个图中，数据的选择和图表类型的确定才是最重要，最需要确定的因素。但是，适当的选择颜色不仅仅能使数据图的阅读者赏心悦目，而且有助于图中数据关系的呈现，使得整个图表更有说服力。这篇文章将简单介绍R语言中的色彩和调色板相关package和函数。

一、R语言中的颜色与调色板

1 R语言中的颜色

R中可以通过定义col参数自定义颜色。参数的设置的四种方法是等价的：数字（如1代表当前palette的第1种颜色，2代表当前palette的第2种颜色等）、颜色名（如”red”,”blue”)、使用rgb()函数得到的返回值或者使用十六进制颜色代码。这几种设置方法是等价的，比如下例：

```
barplot(1:4, col = c(2, "red", rgb(1, 0, 0), "#FF0000"))
```



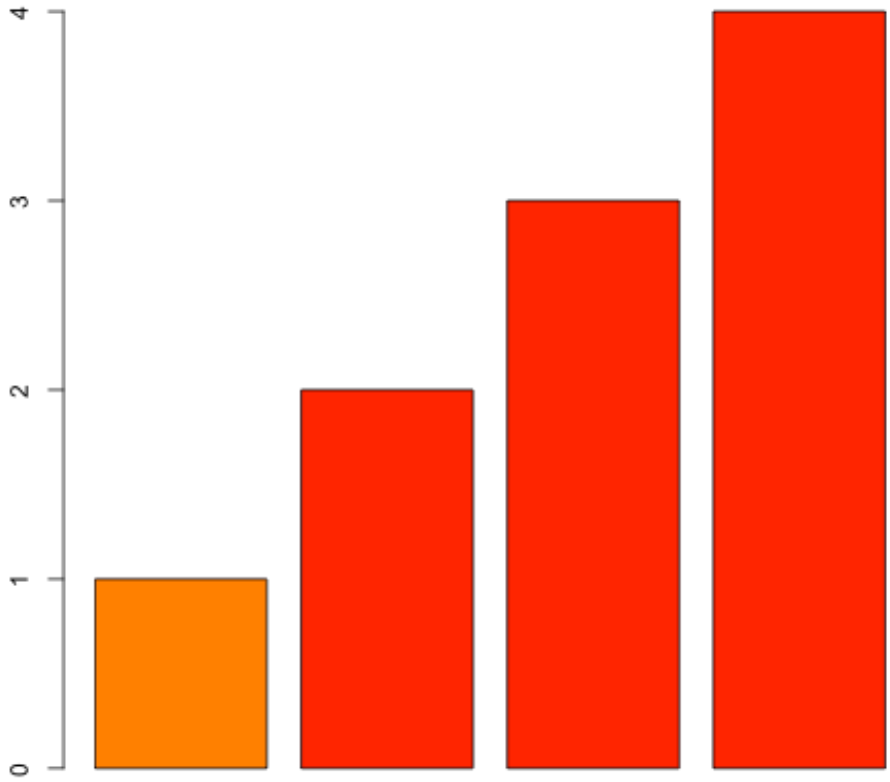
人们不禁要问，不同数字代表的都是什么颜色？通过palette()函数，可以看到在当前调色板下，第一种颜色是黑色，第二种颜色是红色。这个调色板共有8种颜色，当使用颜色数大于8时，会从头开始。

```
palette()


## [1] "black"   "red"     "green3"  "blue"    "cyan"    "magenta" "yellow"
## [8] "gray"
```

调色板当然是可以改变的，比如用系统中的彩虹调色板。此时，第二位可就不是红色了。

```
palette(rainbow(12))
barplot(1:4, col = c(2, "red", rgb(1, 0, 0), "#FF0000"))
```



通过再次将palette设置为”default”，可以得到默认调色板。

```
palette("default")
```

可以看到，用数字来设置颜色非常简单，但是可重复性比较差，可能会因为他人调色板设置的不同而导致颜色不一致。一般是在自己进行探索性数据分析时使用得比较多。
直接用颜色的英文来设置，则没有上述问题。R中可以用英文设置的颜色有657个，可以通过colors()函数查看，或者直接运行demo(“colors”)看示例图。不过我这等没有美术基础的色盲，用得多的还是最简单的”red”,”green”,”blue”。

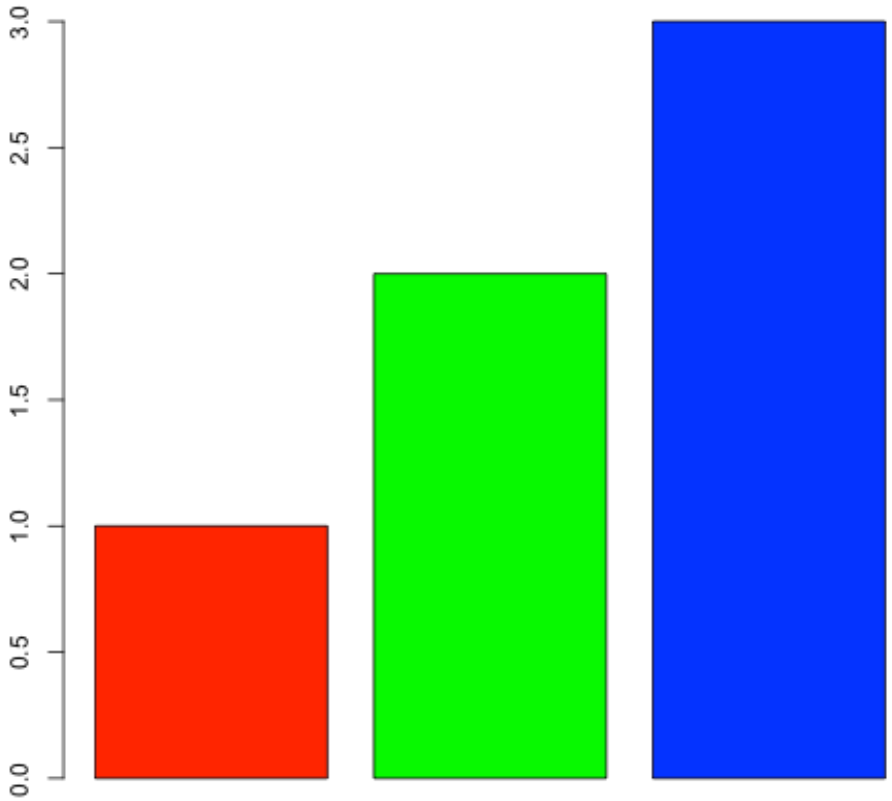
```
head(colors())



## [1] "white"         "aliceblue"     "antiquewhite"  "antiquewhite1"
## [5] "antiquewhite2" "antiquewhite3"
```

RGB函数接入三个取值[0,1]的数字，三个数字分别对应”red”,”green”,”blue”。所以在第一幅图中，通过将”red”设为最大值1，”green”,”blue”设为最小值0，得到的结果就是红色。rgb()函数中参数alpha是用来设置透明度的，对于特别密集的散点图很常用。

```
barplot(1:3, col = c(rgb(1, 0, 0), rgb(0, 1, 0), rgb(0, 0, 1)))
```



其实rgb()函数返回的值就是十六进制颜色代码。所以二者肯定可以换着使了。

```
rgb(1, 0, 0)


## [1] "#FF0000"
```

2 R语言中的调色板

调色板，简单来说就是颜色的列表。其实在上一节中已经出现过一种调色板rainbow了。传入数字12，获得了12个彩虹颜色的颜色列表。传入1000，可以的都下图右侧的彩虹转盘。

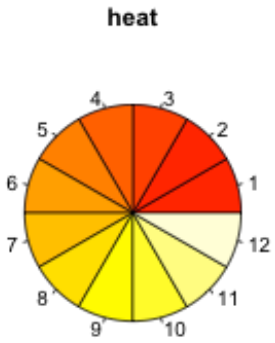
```
par(mfrow = c(1, 2))
pie(rep(1, 12), col = rainbow(12), main = "rainbow12")
pie(rep(1, times = 1000), labels = "", col = rainbow(1000), border = rainbow(1000),
    main = "rainbow1000")
```

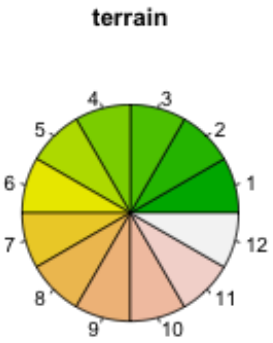
rainbow12

rainbow1000

R语言中除了自带rainbow()调色板还有以下几种:

```
par(mfrow = c(2, 2))
pie(rep(1, 12), col = heat.colors(12), main = "heat")
pie(rep(1, 12), col = terrain.colors(12), main = "terrain")
pie(rep(1, 12), col = topo.colors(12), main = "topo")
pie(rep(1, 12), col = cm.colors(12), main = "cm")
```





二、colorRamp()和colorRampPalette()

colorRamp()和colorRampPalette()都可用于建立颜色板。通过传入希望的主要颜色如蓝、紫，colorRamp()和colorRampPalette都返回一个函数。二者返回的函数区别为：colorRamp()返回的函数像grey()一样，入参为[0,1]之间的数列，数列中数字个数即为函数返回的颜色板色彩数。colorRampPalette()返回的参数则像rainbow()一样，入参为希望返回颜色板色彩的数量。而且通过下例可知，colorRampPalette()返回渐变颜色板函数，而colorRamp()返回对比颜色板函数。虽然都是用同样的颜色，结果不同。

```
par(mfrow = c(1, 2))
b2p1 <- colorRampPalette(c("blue", "purple"))
b2p2 <- colorRamp(c("blue", "purple"))
pie(rep(1, 12), labels = "", col = b2p1(12), border = b2p1(12), main = "colorRampPalette")
pie(rep(1, 12), labels = "", col = b2p2(seq(0, 1, len = 12)), border = b2p2(seq(0,
    1, len = 12)), main = "colorRamp")
```

colorRampPalette

colorRamp

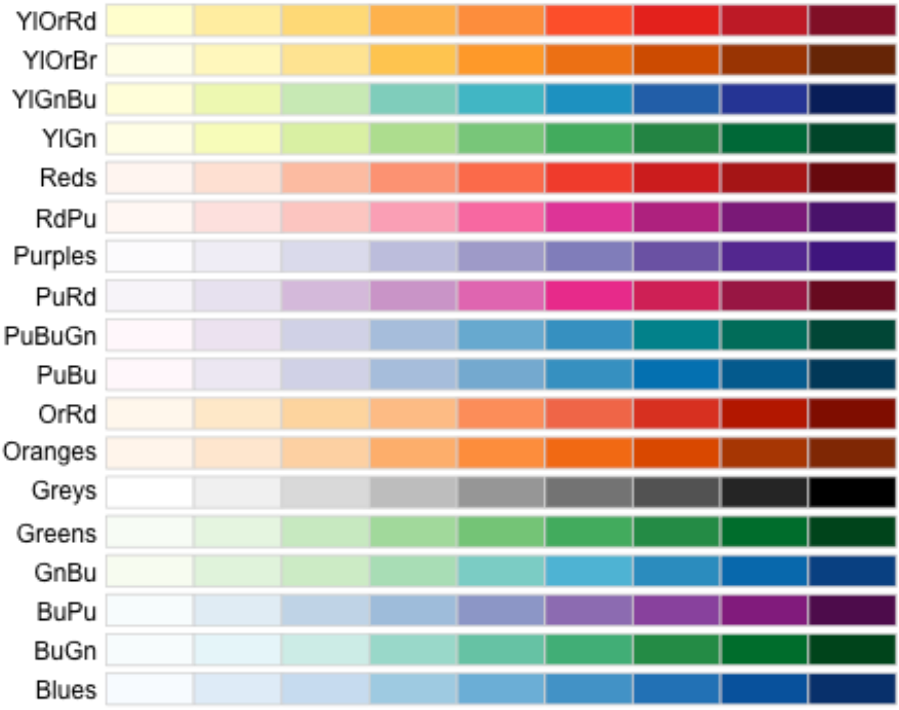
三、RColorBrewer包

虽然说RColorBrewer包中实际用到的就只有brewer.pal()函数，但是包中的两个优点使得其非常实用。一是，包中颜色板被划分为序列型(sequential)、离散型(diverging)、分类型（qualitative)这三种基本能满足统计作图需要的类型；二是，颜色都比较协调。更多指引见其官网ColorBrewer。

```
require("RColorBrewer")
```

序列型颜色板适用于从低到高排序明显的数据，浅色数字小，深色数字大。

```
display.brewer.all(type = "seq")
```

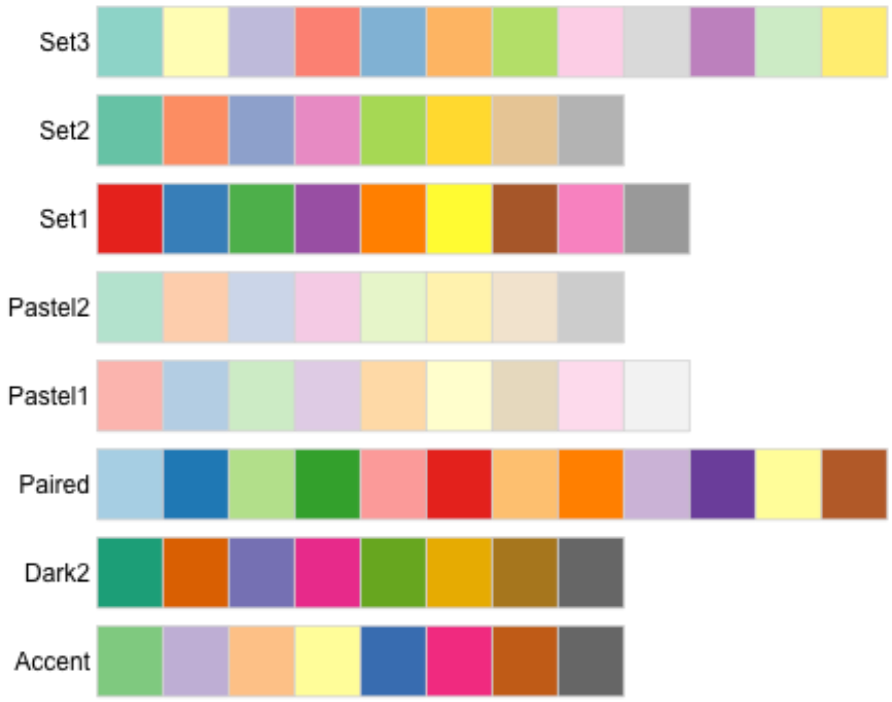


离散型颜色板适合带“正、负”的，对极值和中间值比较注重的数据。

```
display.brewer.all(type = "div")
```

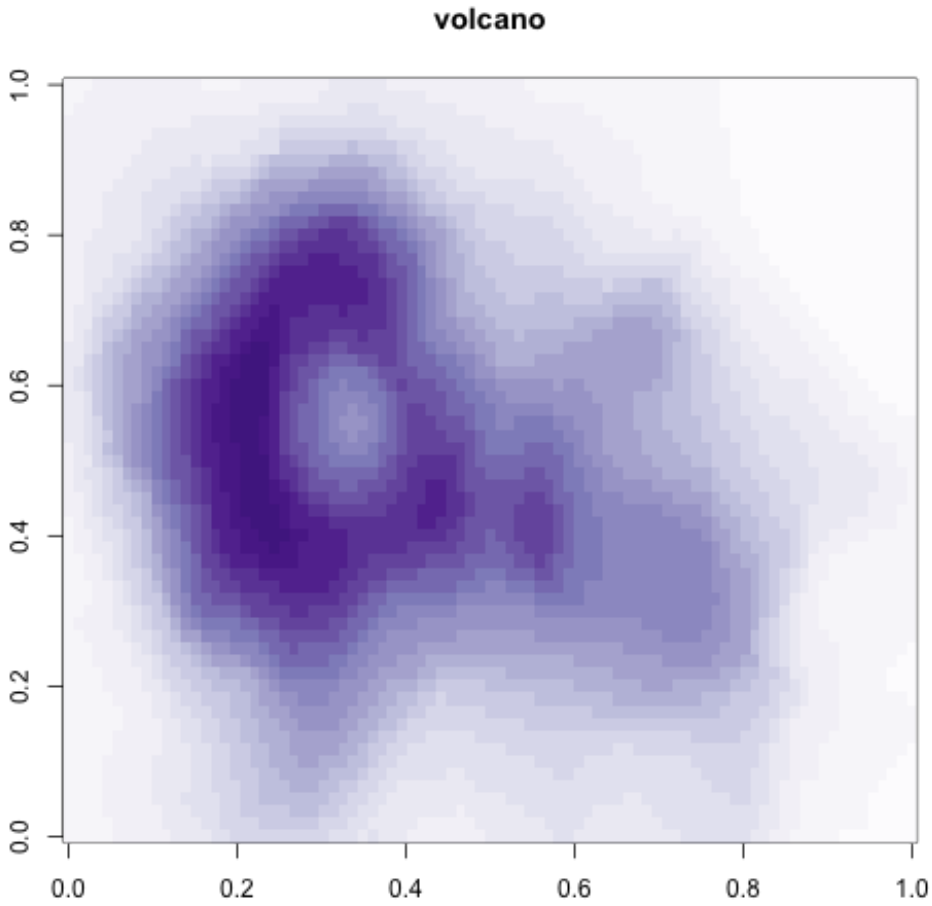
分类型颜色板比较适合区分分类型的数据。

```
display.brewer.all(type = "qual")
```

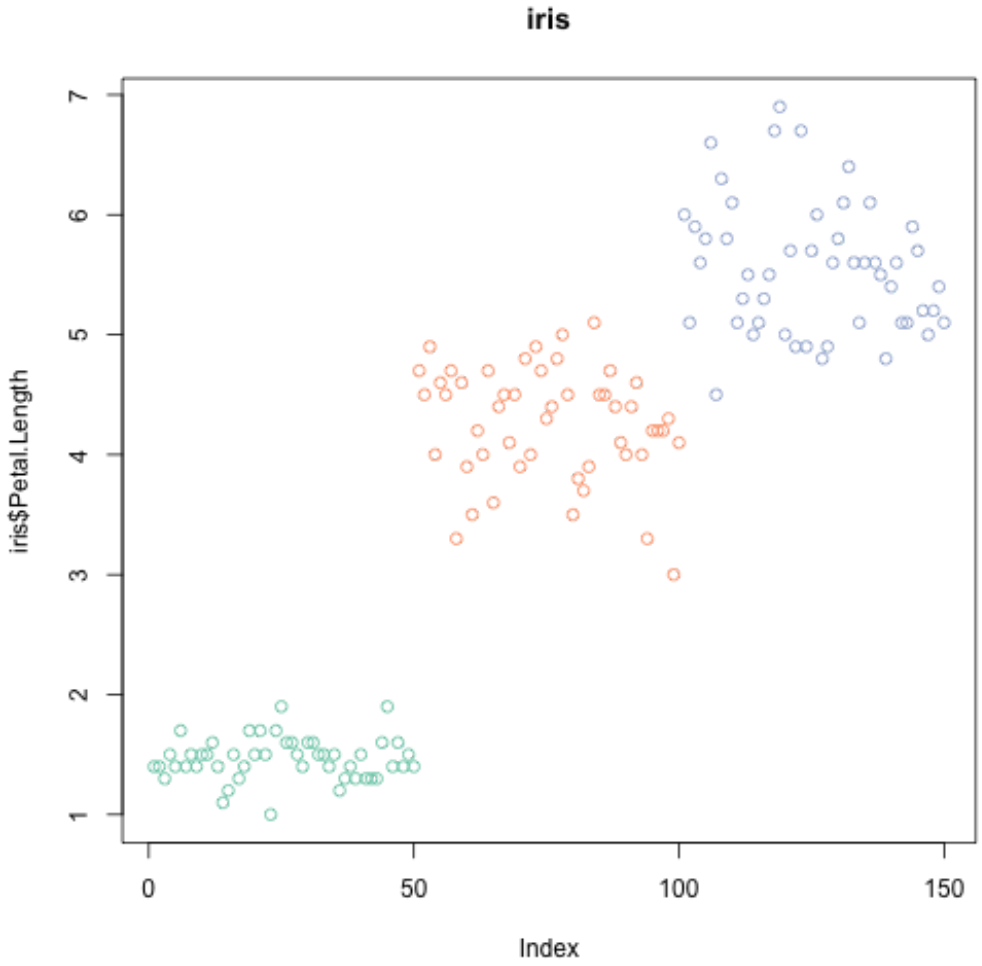


下面两个图是不是比默认颜色好看一些？再不好看的话只能靠自己的艺术细胞，或者从别人的图片上找灵感了。比如说上次文章中写到的从dribble网站上扒下来调色板颜色，可以回忆一下。有更高要求的同学可以参考哈佛的CS171可视化课程，用的是D3。

```
image(volcano, col = colorRampPalette(brewer.pal(9, "Purples"))(20), main = "volcano")
```




```
plot(iris$Petal.Length, iris$Petal.With, col = brewer.pal(3, "Set2")[iris$Species],
    main = "iris")
```



社交帐号登录: 微博 QQ 人人 豆瓣 更多»

说点什么吧…

发布

0条评论 最新 最早 最热

还没有评论，沙发等你来抢

iccm正在使用多说

声明: 本文采用 BY-NC-SA 授权。转载请注明转自: CCM

« 分类器评价与在R中的实现：ROC图与AUC

动态局部回归过程 »

Copyright © 2014 i CCM All rights reserved.