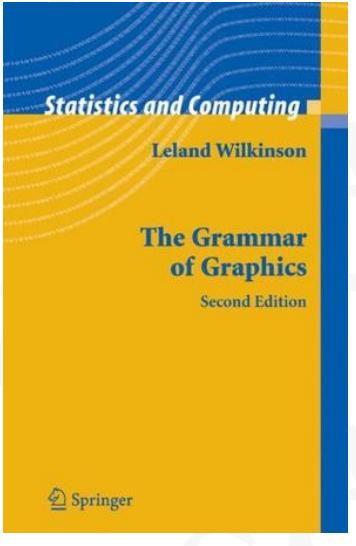
R图形------ggplot2

陈华珊 (社会发展战略研究院)

2015-10-25

优点

- 采用图层的设计方式
- 是一套关于控制图形的语法
- 避免繁琐细节
- 得益于 Leland Wilkinson 在他的著作《The Grammar of Graphics》中提出了一套图形语法，把图形元素抽象成可以自由组合的成分，Hadley Wickham 把这套想法在 R 中实现。



ggplot2的基本概念

◆ 数据（Data）和映射（Mapping）

将数据中的变量映射到图形属性。映射控制了二者之间的关系。

length	width	depth	trt
2	3	4	a
1	2	1	a
4	5	15	b
9	10	80	b

x	y	colour
2	3	a
1	2	a
4	5	b
9	10	b

◆ 标度（Scale）

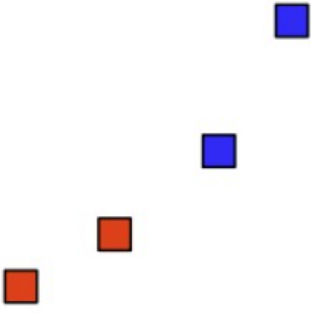
标度负责控制映射后图形属性的显示方式。具体形式上来看是图例和坐标刻度。Scale和Mapping是紧密相关的概念。

x	y	colour
2	3	a
1	2	a
4	5	b
9	10	b

x	y	colour
25	11	red
0	0	red
75	53	blue
200	300	blue

◆ 几何对象（Geometric）

几何对象代表我们在图中实际看到的图形元素，如点、线、多边形等。



Geoms

❖ ggplot 中的几何对象类型

```
require(ggplot2)
params <- ls(pattern = '^geom_', env = as.environment('package:ggplot2'))
gsub("geom_", "", params)
##  [1] "abline"     "area"       "bar"        "bin2d"      "blank"
##  [6] "boxplot"    "contour"    "crossbar"   "density"    "density2d"
## [11] "dotplot"    "errorbar"   "errorbarh"  "freqpoly"   "hex"
## [16] "histogram"  "hline"      "jitter"     "line"       "linerange"
## [21] "map"        "path"       "point"      "pointrange" "polygon"
## [26] "quantile"   "raster"     "rect"       "ribbon"     "rug"
## [31] "segment"    "smooth"     "step"       "text"       "tile"
## [36] "violin"     "vline"
```

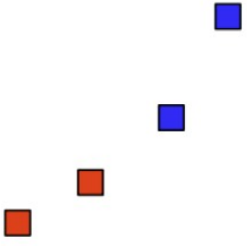
◆ 统计变换（Statistics）

对原始数据进行某种计算，例如对二元散点图加上一条回归线。

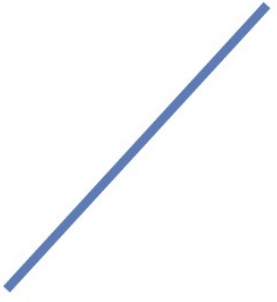
Geoms　　Stat

◆ 坐标系统（Coordinate）

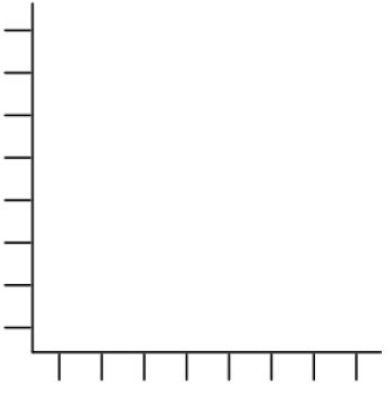
坐标系统控制坐标轴开影响所有图形元素，坐标轴可以进行变换以满足不同的需要。



Geoms



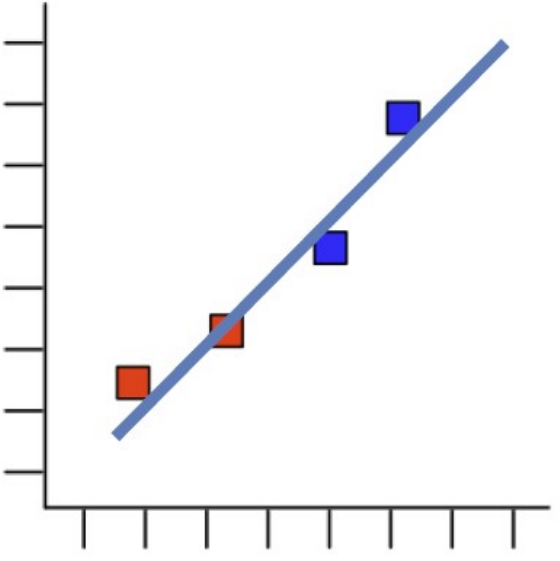
Stat



Coord

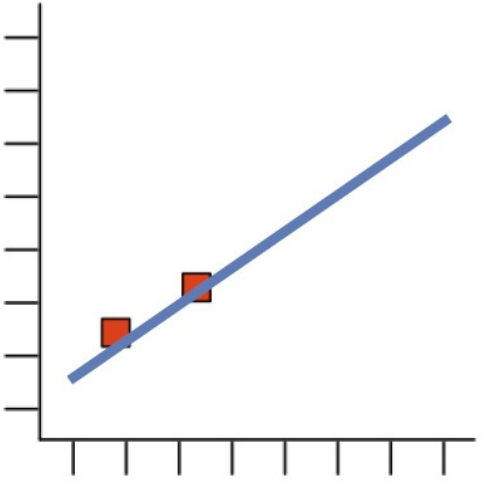
◆ 图层（Layer）

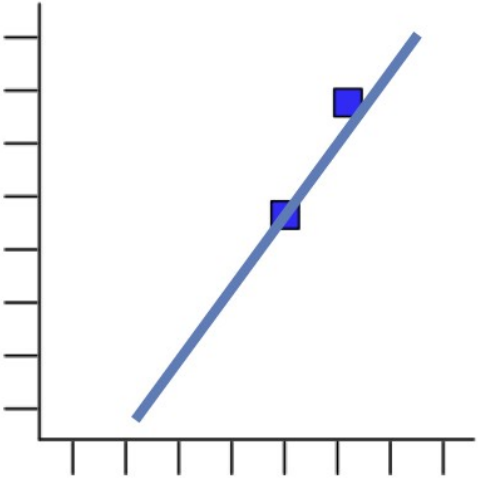
数据、映射、几何对象、统计变换等构成一个图层。图层可以允许用户一步步的构建图形，方便单独对图层进行修改。



◆ 分面（Facet）

创建一系列图形。将数据按某种方式分组，然后分别绘图。分面就是控制分组绘图的方法和排列形式。

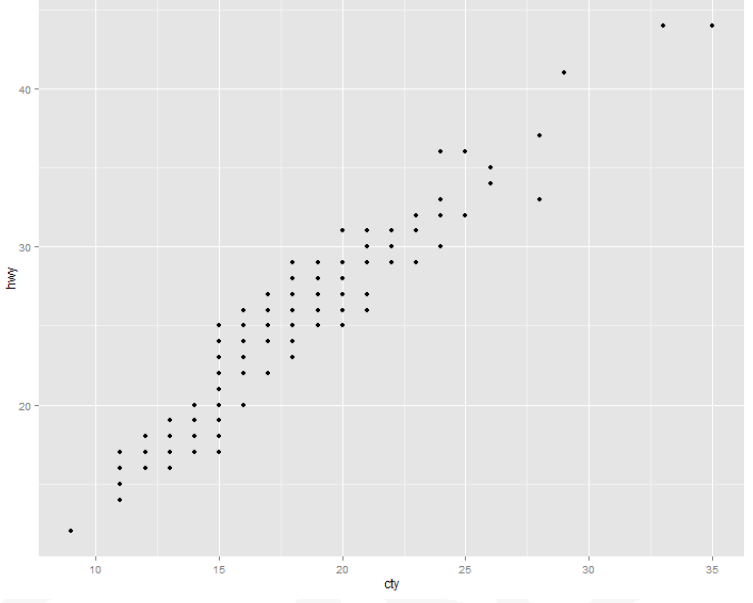




示例

◆ 散点图

```
1  library(ggplot2)
2  p <- ggplot(data = mpg, mapping = aes(x = cty, y = hwy))
3  p + geom_point()
```

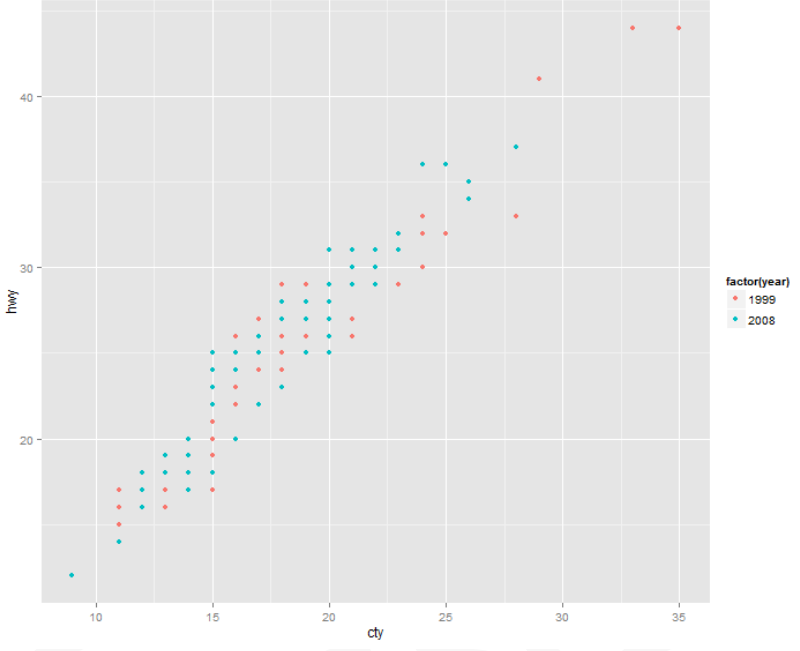


❖ 查看属性

```
summary(p)
## data: manufacturer, model, displ, year, cyl, trans, drv, cty, hwy,
##   fl, class [234x11]
## mapping:  x = cty, y = hwy
## faceting: facet_null()
summary(p+geom_point())
## data: manufacturer, model, displ, year, cyl, trans, drv, cty, hwy,
##   fl, class [234x11]
## mapping:  x = cty, y = hwy
## faceting: facet_null()
## -----------------------------------
## geom_point: na.rm = FALSE
## stat_identity:
## position_identity: (width = NULL, height = NULL)
```

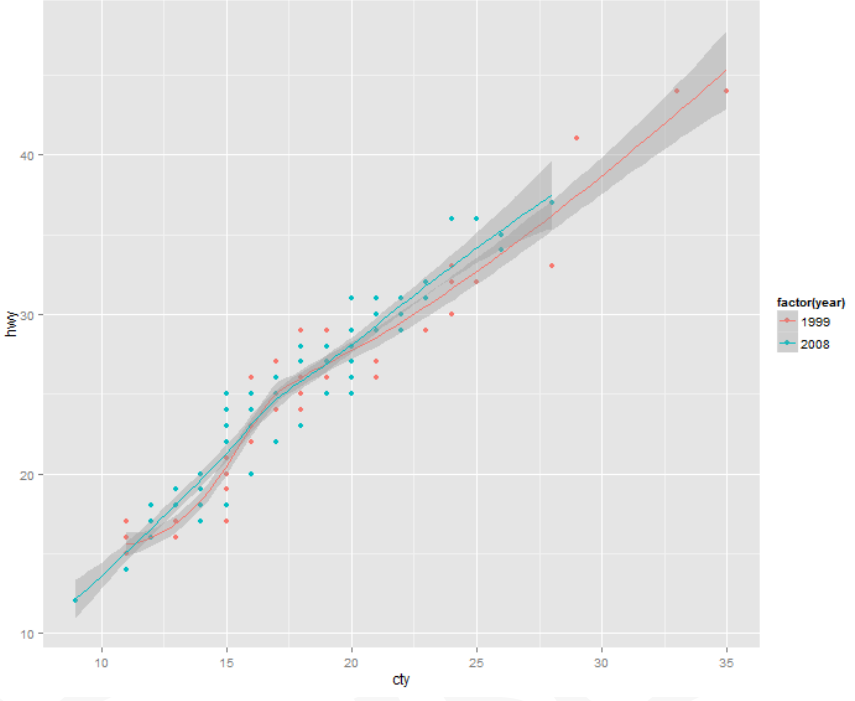
❖ 将年份映射到颜色属性

```
1    p <- ggplot(mpg, aes(x = cty, y = hwy, colour = factor(year)))
2    p + geom_point()
```



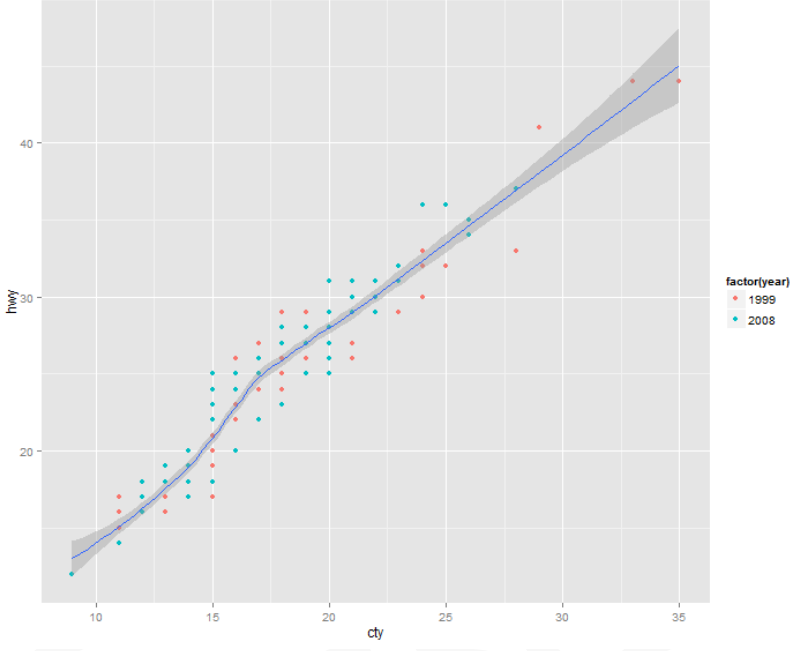
❖ 增加平滑曲线，两条平滑曲线

```
1    p + geom_point() + stat_smooth()
```



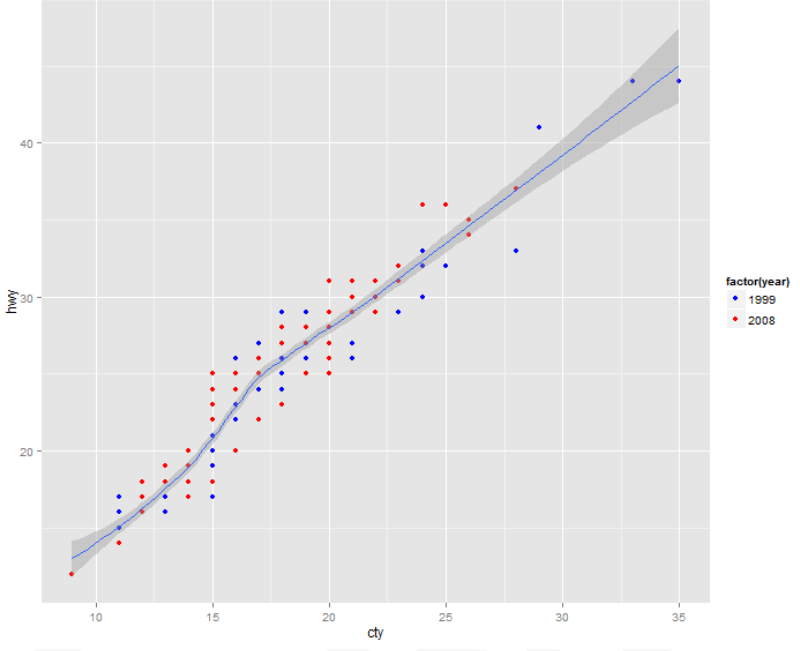
❖ 一条平滑曲线

```
1    p <- ggplot(mpg, aes(x=cty, y=hwy))
2    p + geom_point(aes(colour = factor(year))) + stat_smooth()
```



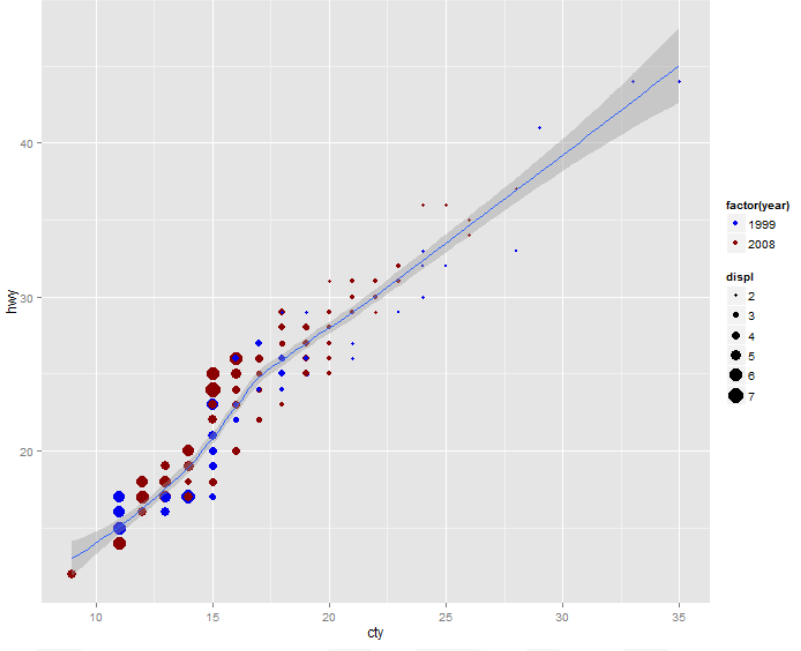
❖ 用标度来修改颜色取值

```
1    p + geom_point(aes(colour = factor(year))) + stat_smooth()+
2      scale_color_manual(values =c('blue','red'))
```



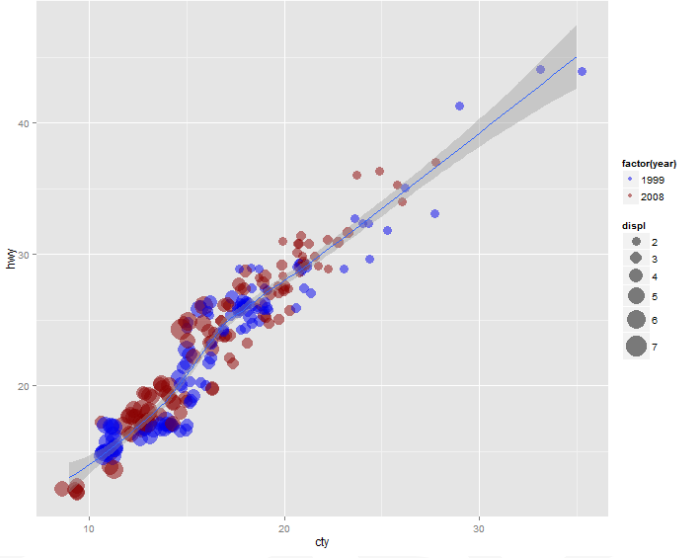
❖ 用散点大小表示车排量

```
1    p + geom_point(aes(colour=factor(year),size=displ)) + stat_smooth()+
2      scale_color_manual(values =c('blue2','red4'))
```



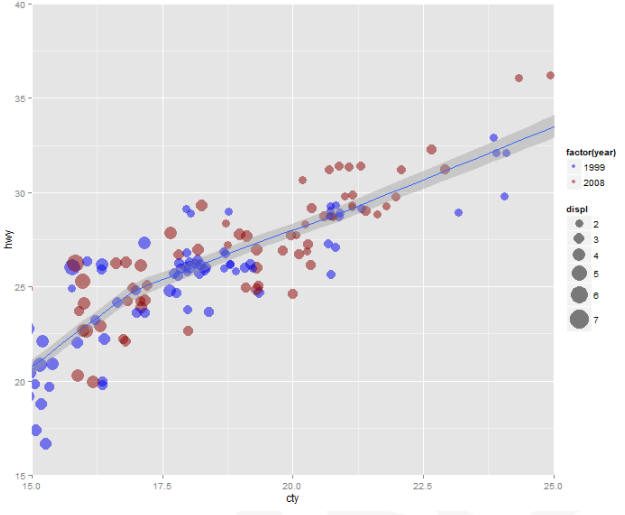
❖ 调整大小及颜色

```
1   p + geom_point(aes(colour=factor(year),size=displ), alpha=0.5,position
    = "jitter") + stat_smooth() +
2     scale_color_manual(values =c('blue2','red4'))+
3     scale_size_continuous(range = c(4, 10))
```



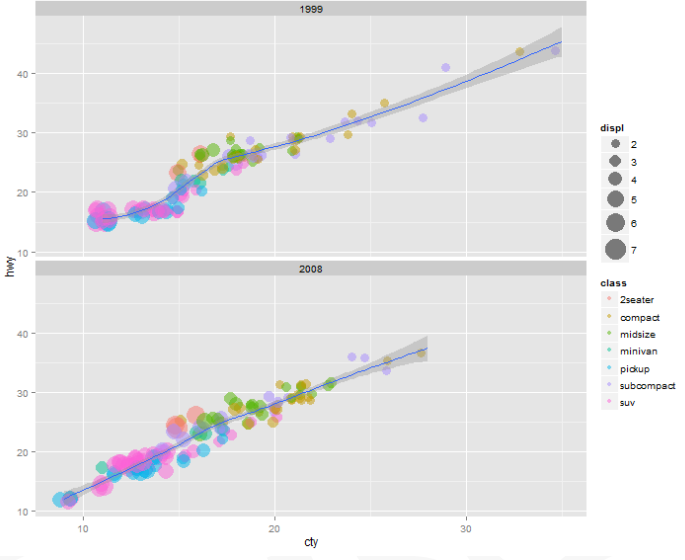
❖ 用坐标控制图形显示的范围

```
1   p + geom_point(aes(colour=factor(year),size=displ), alpha=0.5,position
    = "jitter") + stat_smooth() +
2     scale_color_manual(values =c('blue2','red4')) +
3     scale_size_continuous(range = c(4, 10)) +
4     coord_cartesian(xlim = c(15, 25), ylim=c(15,40))
```



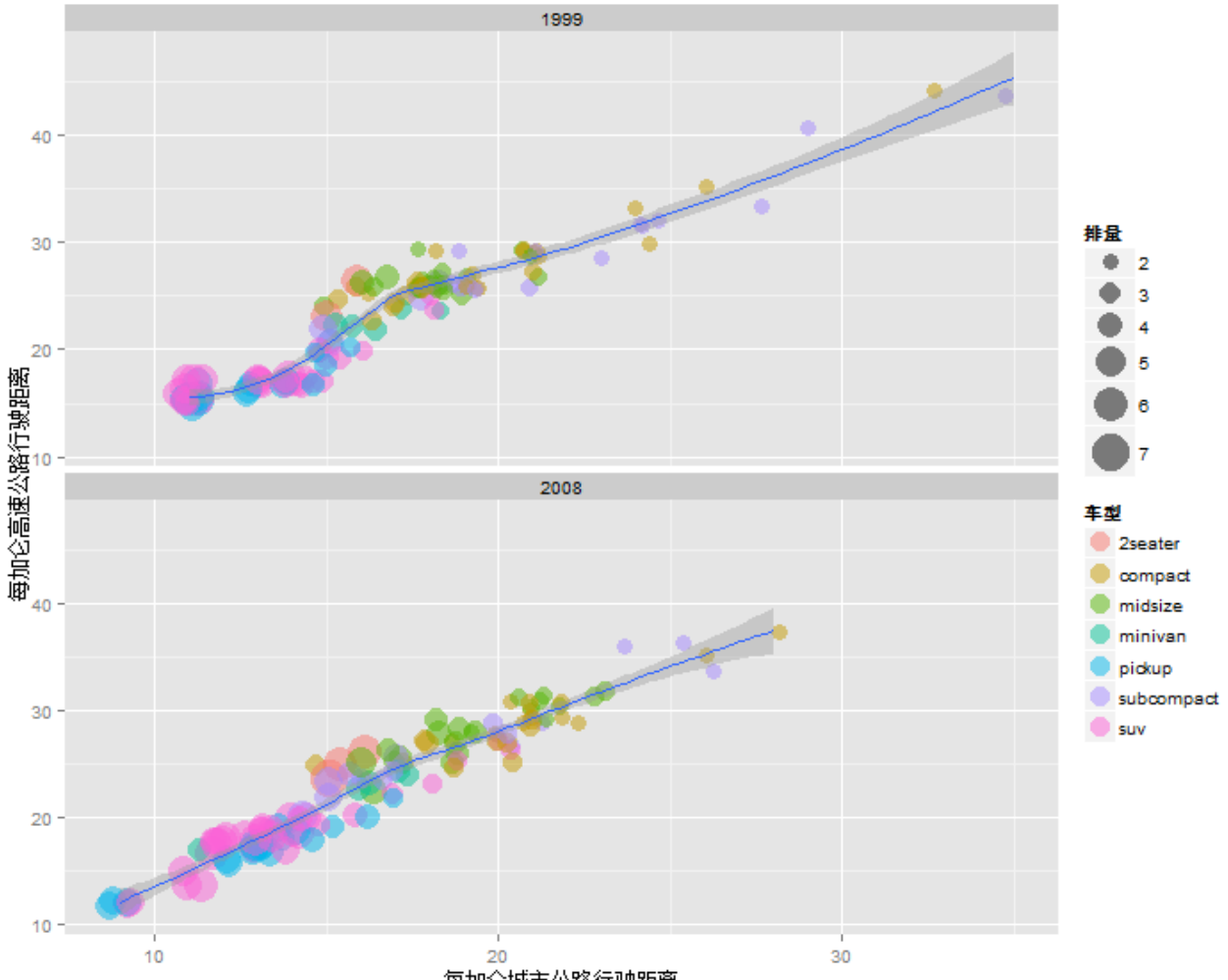
❖ 利用facet分别显示不同年份的数据

```
1   p + geom_point(aes(colour=class, size=displ) , alpha=0.5, position
    = "jitter") + stat_smooth()+
2     scale_size_continuous(range = c(4, 10))+
3     facet_wrap(~ year, ncol=1)
```



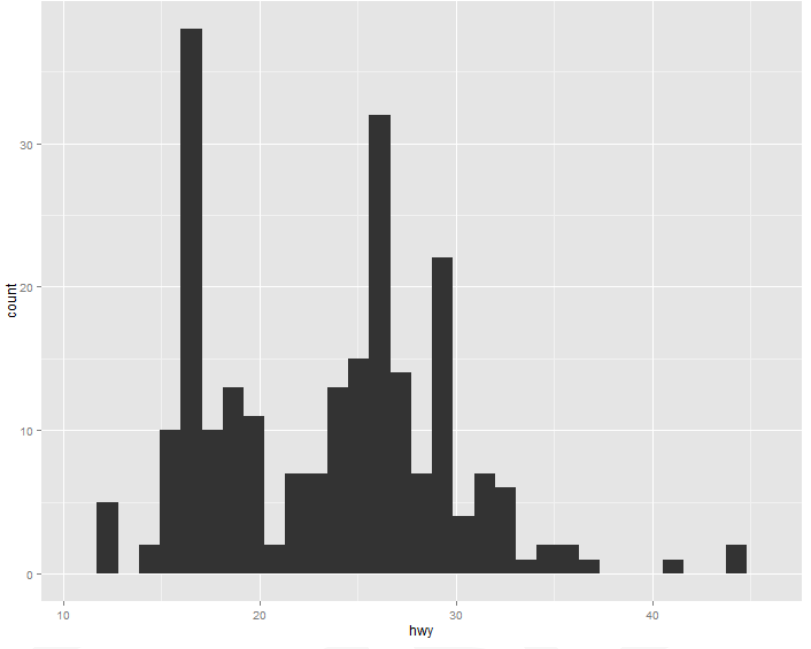
❖ 增加图名并精细修改图例

```
1  p <- ggplot(mpg, aes(x=cty, y=hwy))
2  p + geom_point(aes(colour=class,size=displ) , alpha=0.5,position =
   "jitter") +
3    stat_smooth()+
4    scale_size_continuous(range = c(4, 10))+
5    facet_wrap(~ year, ncol=1)+
6    labs(y='每加仑高速公路行驶距离', x='每加仑城市公路行驶距离')+
7    guides(size=guide_legend(title='排量'), colour = guide_legend(title='车
   型',
8       override.aes=list(size=5)))
```


1999
2008
每加仑高速公路行驶距离
每加仑城市公路行驶距离
10
20
30
40
排量
2
3
4
5
6
7
车型
2seater
compact
midsize
minivan
pickup
subcompact
suv

◆ 直方图

```
1    p <- ggplot(mpg, aes(x=hwy))
2    p + geom_histogram()
```



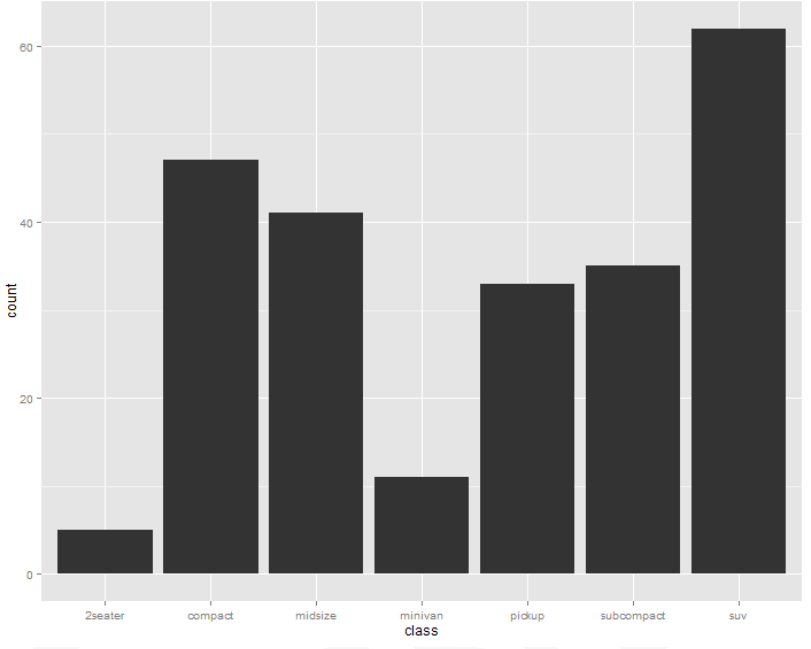
❖ 直方图的几何对象中内置有默认的统计变换

```
summary(p + geom_histogram())
## data: manufacturer, model, displ, year, cyl, trans, drv, cty,
hwy,
##   fl, class [234x11]
## mapping:  x = hwy
## faceting: facet_null()
## ----------------------------------
## geom_histogram:
## stat_bin:
## position_stack: (width = NULL, height = NULL)

p + geom_histogram(aes(fill=factor(year)  ,y=.  .density.  .  )  ,
alpha=0.3,colour='black')+
  stat_density(geom='line',position='identity',size=1.5, aes(colour=facto
  facet_wrap(~year,ncol=1)
```

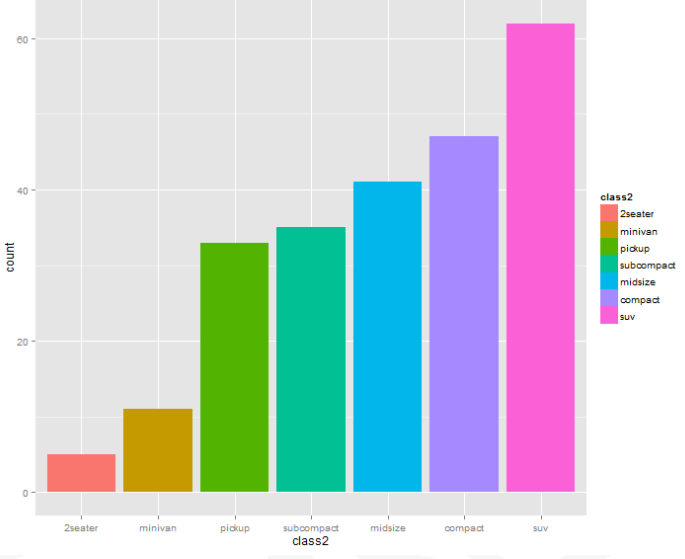
◆ 条形图

```
1    p <- ggplot(mpg, aes(x=class))
2    p + geom_bar()
```



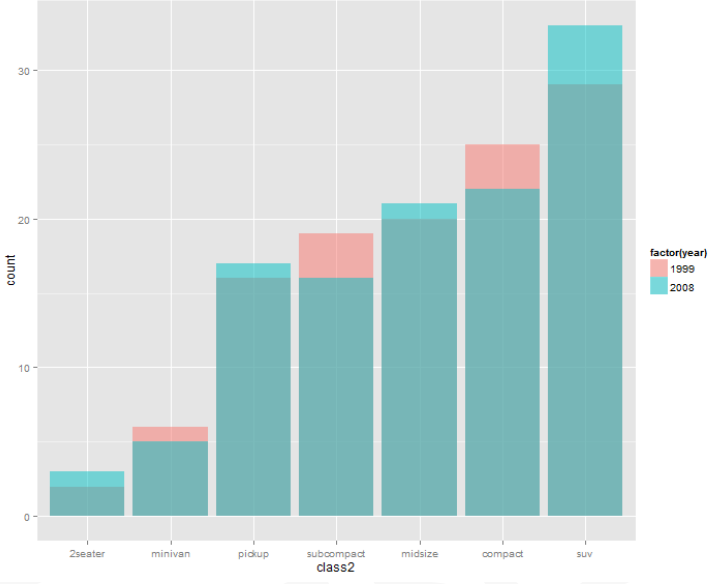
❖ 将计数排序后绘制条形图

```
1  class2 <- mpg$class; class2 <- reorder(class2,class2,length)
2  mpg$class2 <- class2
3  p <- ggplot(mpg, aes(x=class2))
4  p + geom_bar(aes(fill=class2))
```



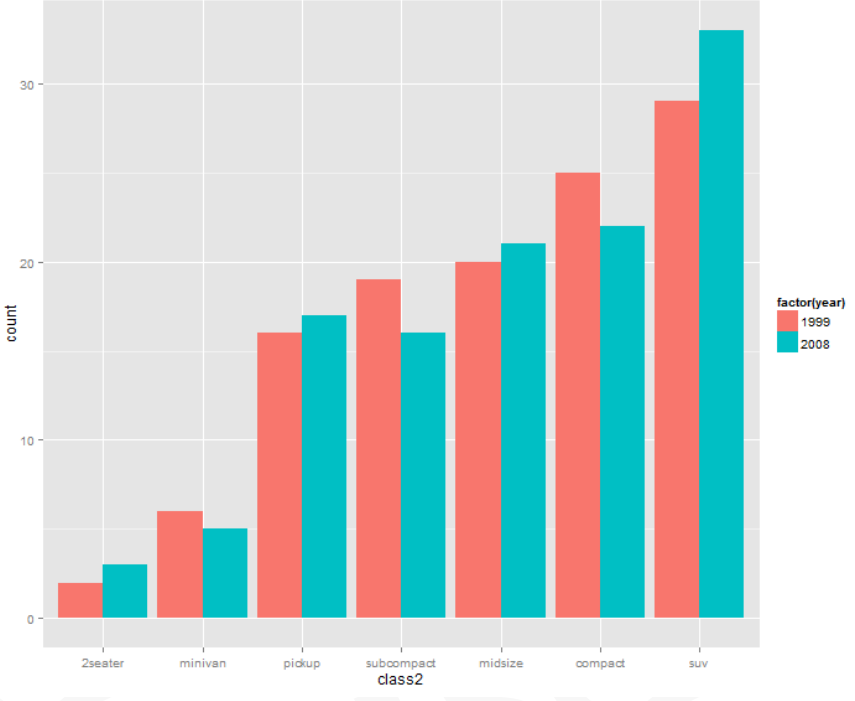
- 根据年份分别绘制条形图，position 控制位置调整方式。

```
1  p <- ggplot(mpg, aes(class2, fill=factor(year)))
2  p + geom_bar(position='identity', alpha=0.5)
```



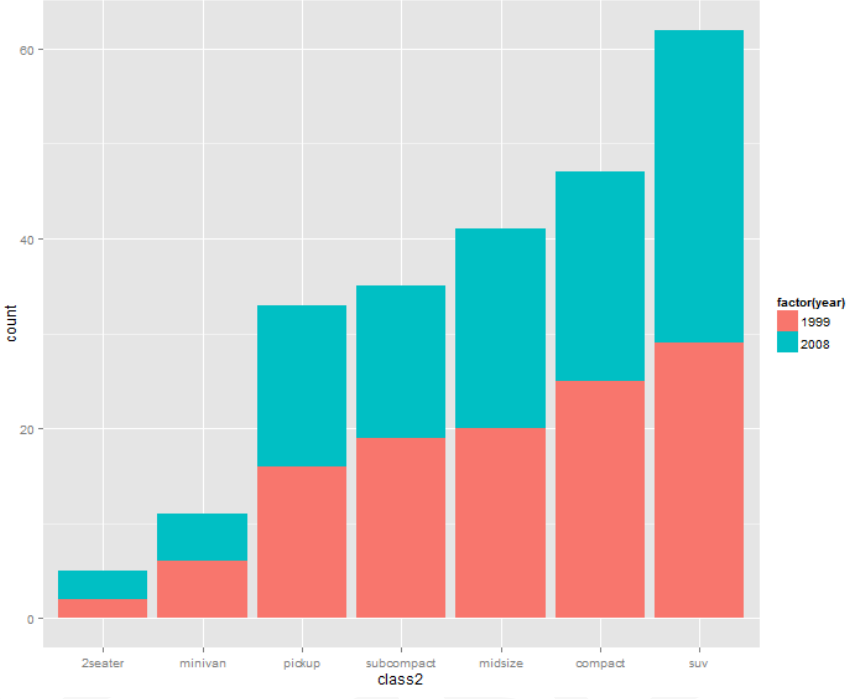
❖ 并立方式

```
1    p + geom_bar(position='dodge')
```



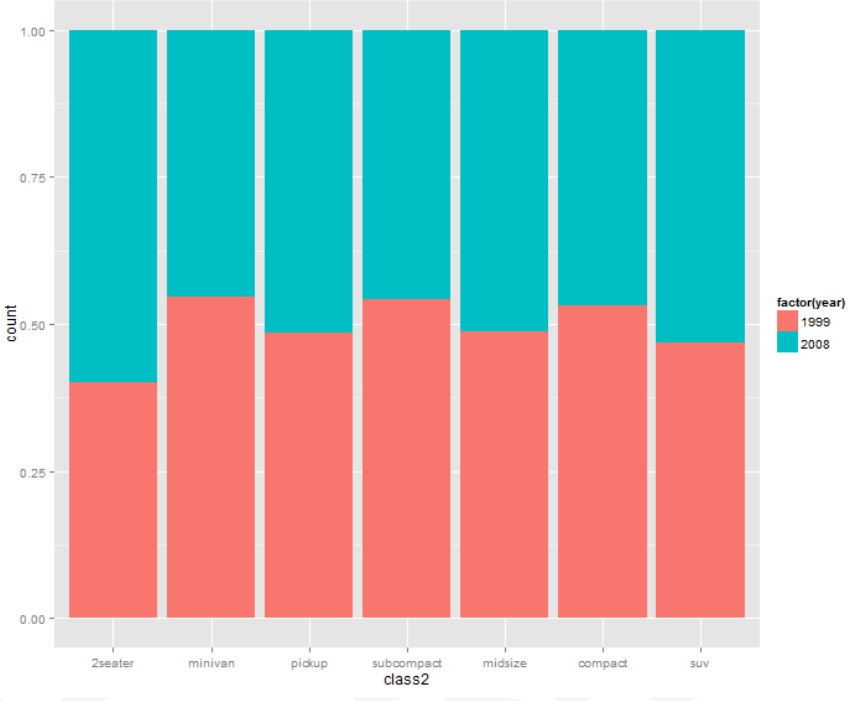
❖ 叠加方式（默认）

```
1    p + geom_bar(position='stack')
```



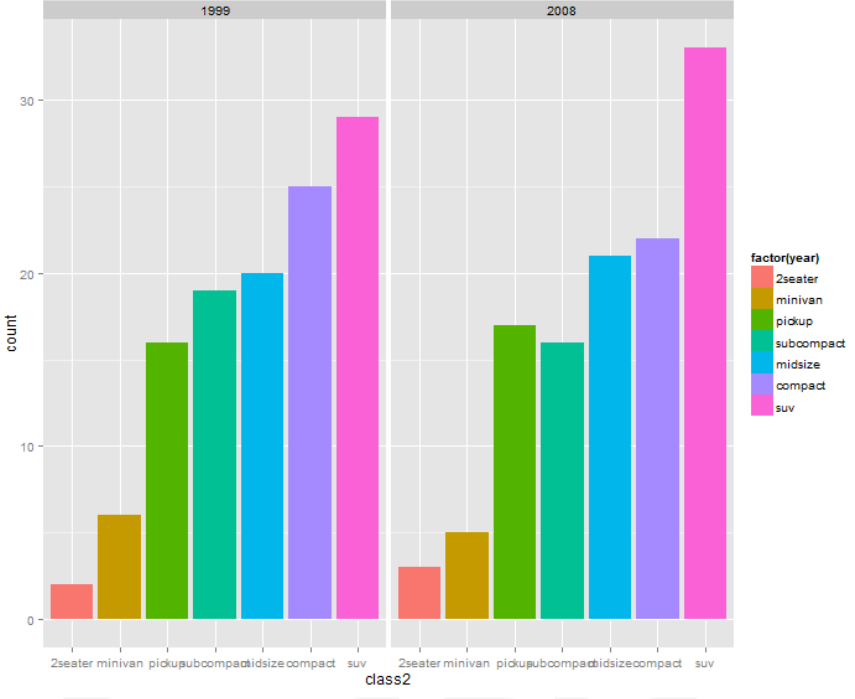
❖ 相对比例

```
1    p + geom_bar(position='fill')
```



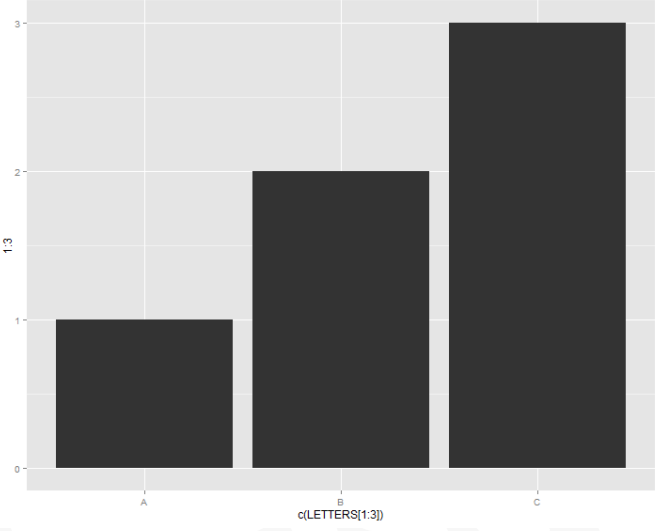
❖ 分面显示

```
1    p + geom_bar(aes(fill=class2))+facet_wrap(~year)
```



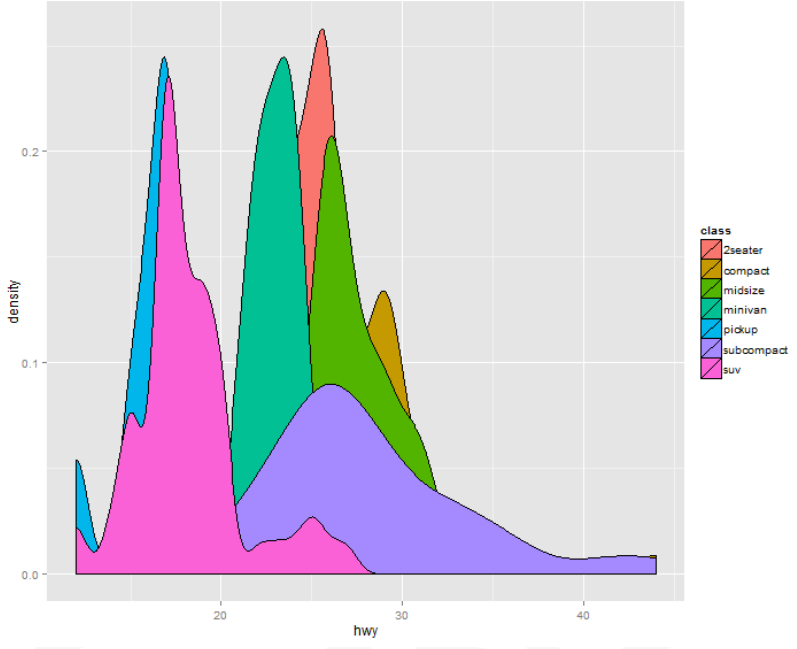
柱状图两个要素，一个是分类变量，一个是数目，也就是柱子的高度。在上面这些例子中都只使用了 X 轴的映射，没有提供 Y 轴的映射，ggplot 会自动计算 class2 各个类别的计数并映射到 Y 轴上。通过 stat 参数，可以让 geom_bar 按指定高度画图，比如：

```
1  ggplot( ) + geom_bar(aes(x = c(LETTERS[1:3] ) , y = 1:3) , stat =
   "identity")
```



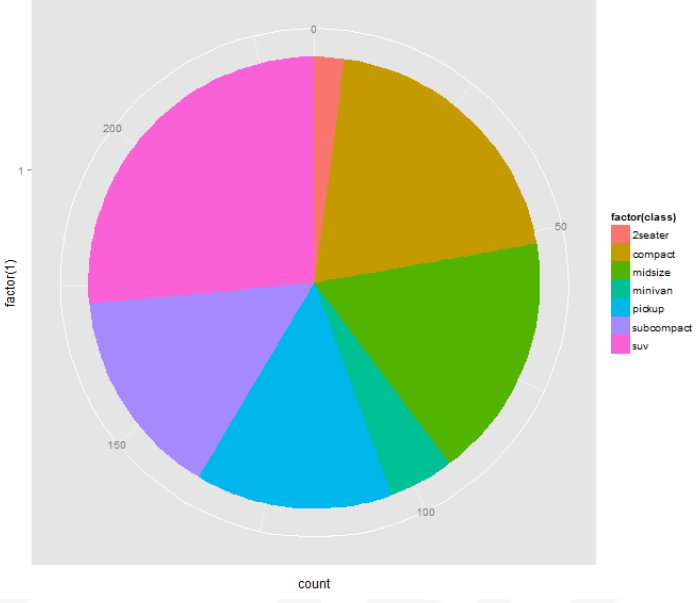
◆ 密度图

```
1    p <- ggplot(mpg)
2    p + geom_density(aes(x=hwy, fill = class))
```



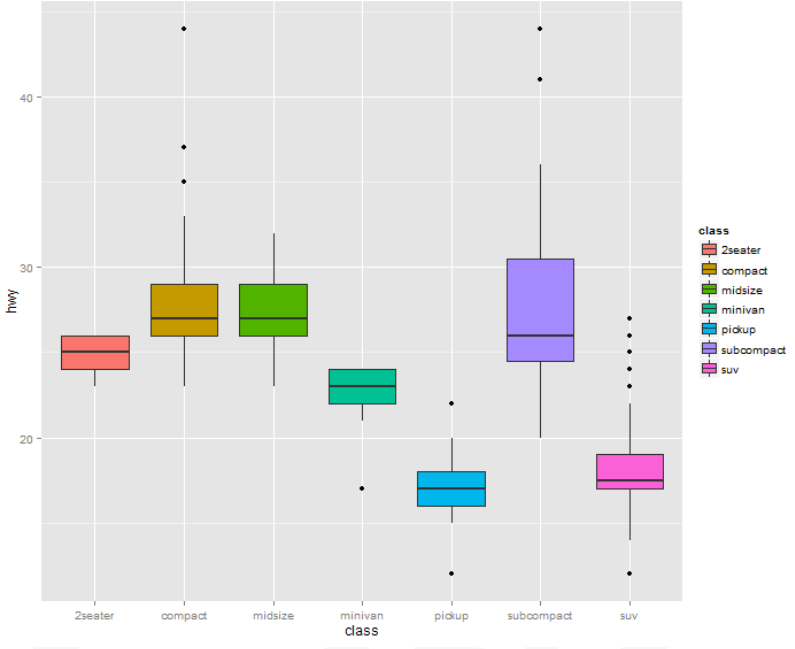
◆ 饼图

```
1    p <- ggplot(mpg, aes(x = factor(1), fill = factor(class))) +
2      geom_bar(width = 1)
3    p + coord_polar(theta = "y")
```



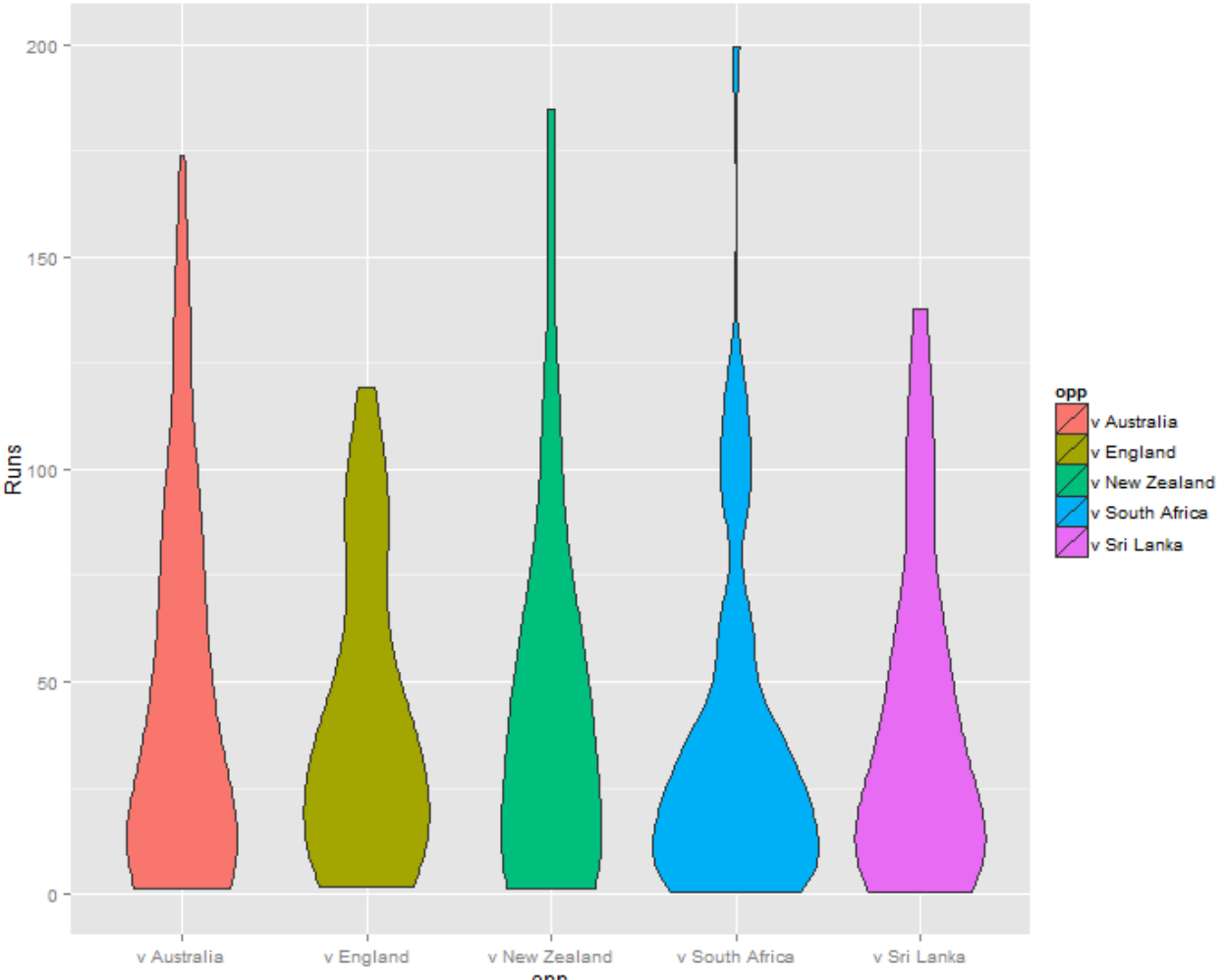
◆ 箱线图

```
1    p <- ggplot(mpg, aes(class,hwy,fill=class))
2    p + geom_boxplot()
```

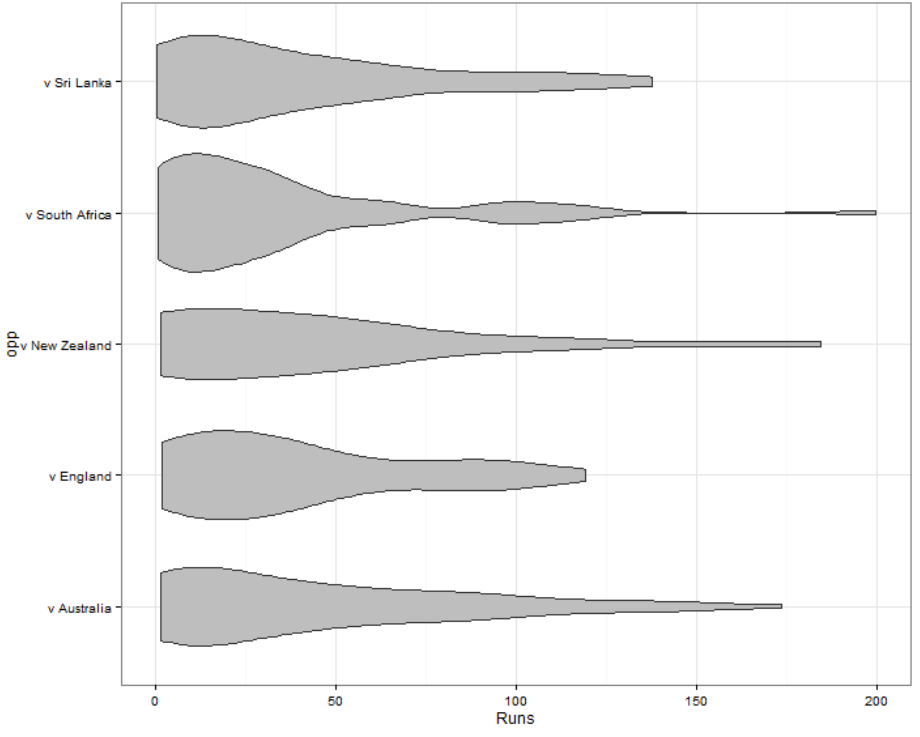


◆ 小提琴图

```
1   s10=read.csv(file=file.path('.','data', 'sachin.csv'))
2
3   s10c=subset(s10,Opposition%in%c("v Australia","v England","v South
    Africa","v New Zealand","v Sri Lanka"))
4
5   s10c$opp = s10c$Opposition
6
7   ####violinplots:RunsvsCountries
8   p1=ggplot(s10c,aes(opp,Runs))
9   p2=p1+geom_violin(aes(fill=opp))
10  print(p2)
```

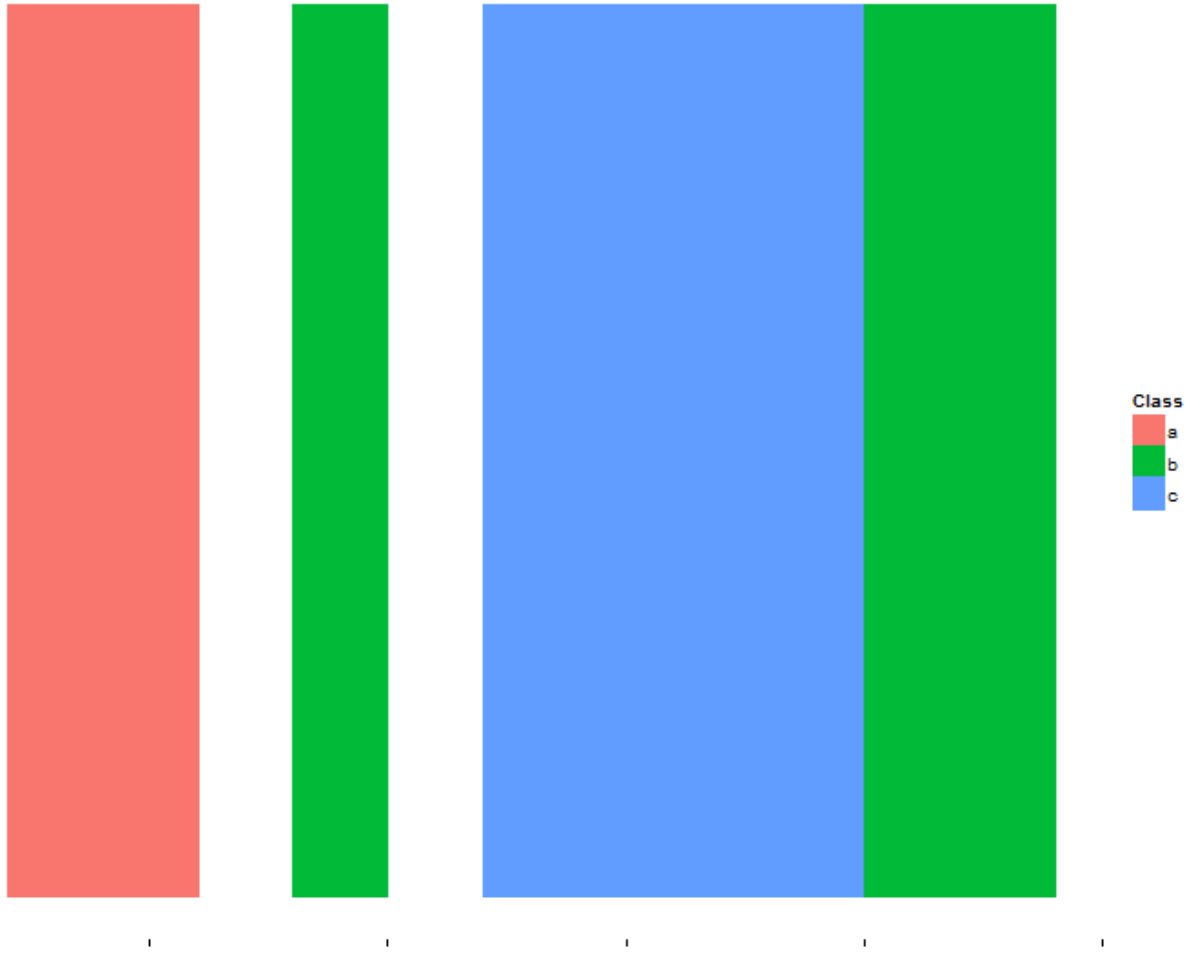

200
150
100
50
0
Runs
v Australia
v England
v New Zealand
v South Africa
v Sri Lanka
opp
opp
v Australia
v England
v New Zealand
v South Africa
v Sri Lanka

```
1    p3 = p1+geom_violin(fill=I("gray")) + coord_flip() + theme_bw()
2    print(p3)
```



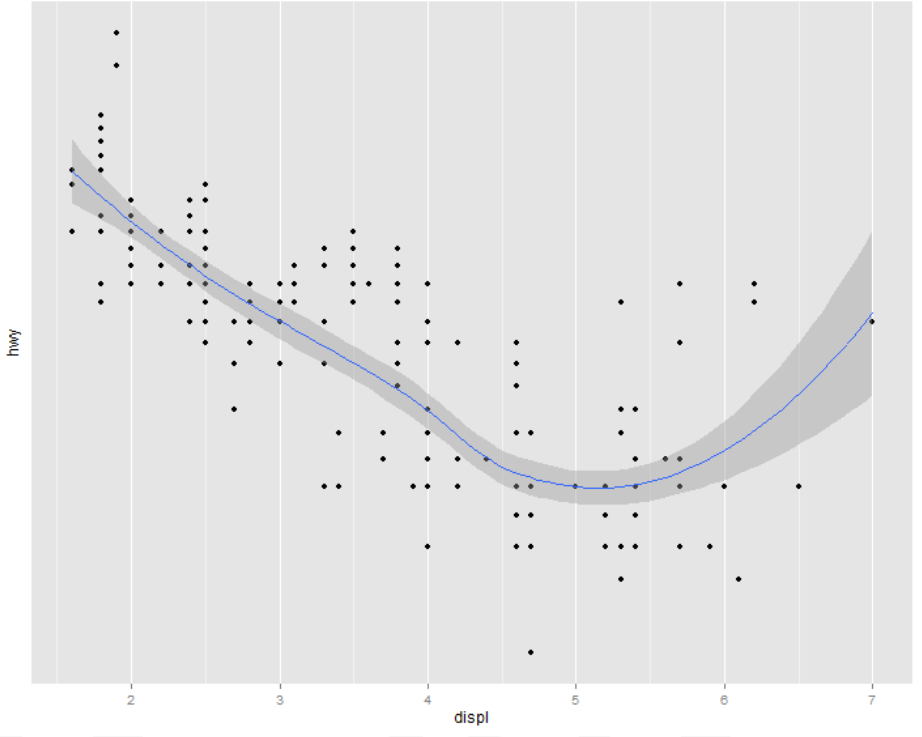
◆ 方块图

```
1   library(ggplot2)
2   DF <- read.table(text="From  To  Class
3   1     3   a
4   4     5   b
5   6     10  c
6   10    12  b", header=TRUE)
7
8   ggplot(DF, aes(xmin=From, xmax=To, ymin=0, ymax=1, colour=Class,
    fill=Class)) +
9     geom_rect() +
10    theme_minimal() +
11    theme(axis.title=element_blank(),
12          axis.text.y=element_blank(),
13          axis.ticks.y=element_blank(),
14          axis.line.y=element_blank(),
15          panel.grid=element_blank())
```


Class
a
b
c

统计变换

```
1    ggplot(mpg, aes(x=displ, y=hwy)) + geom_point() + scale_y_log10()
     + stat_smooth()
```



◆ 在何处设置映射参数

aes 所提供的映射参数设置在 ggplot() 中，而不是提供给 geom_point()，因为:

- ggplot() 里的参数，相当于全局变量，geom_point() 和 stat_smooth() 都知道 x，y 的映射。
- 在 geom_point() 中设置映射参数，则相当于局部变量，geom_point 知道这种映射，而 stat_smooth 不知道。

◆ ggplot 中提供的统计变换方式

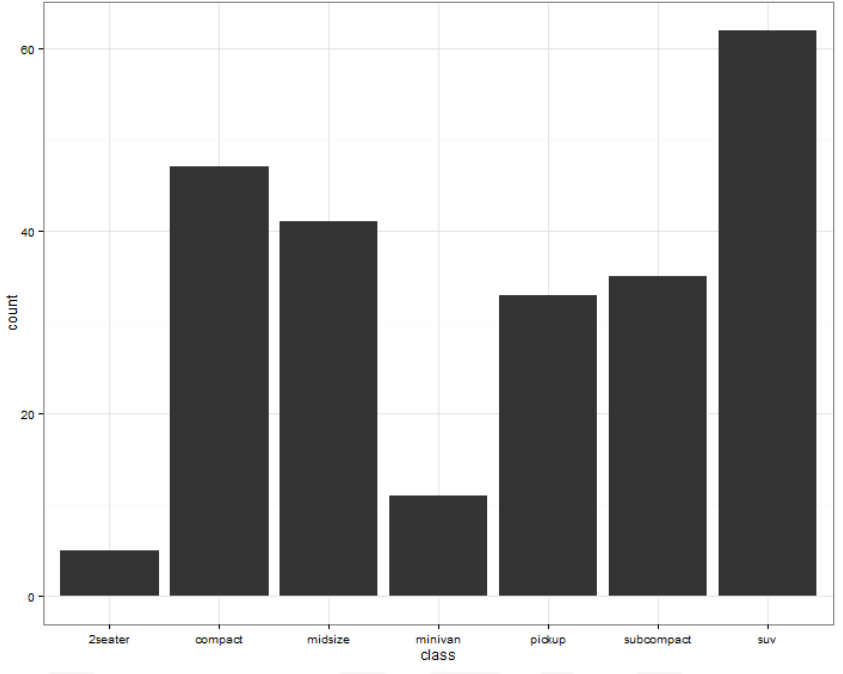
统计变换是非常重要的功能，我们可以自己写函数，基于原始数据做某种计算，并在图上表现出来，也可以通过它改变 geom_xxx() 函数画图的默认统计参数。

```
ls(pattern = '^stat_', env = as.environment('package:ggplot2'))
##  [1] "stat_abline"      "stat_bin"         "stat_bin2d"
##  [4] "stat_bindot"      "stat_binhex"      "stat_boxplot"
##  [7] "stat_contour"     "stat_density"     "stat_density2d"
## [10] "stat_ecdf"        "stat_ellipse"     "stat_function"
## [13] "stat_hline"       "stat_identity"    "stat_qq"
## [16] "stat_quantile"    "stat_smooth"      "stat_spoke"
## [19] "stat_sum"         "stat_summary"     "stat_summary_hex"
## [22] "stat_summary2d"   "stat_unique"      "stat_vline"
## [25] "stat_ydensity"
```

主題

通过 `theme()` 函数对外观进行更精准的调整，`ggplot2` 默认设置了一些主题:

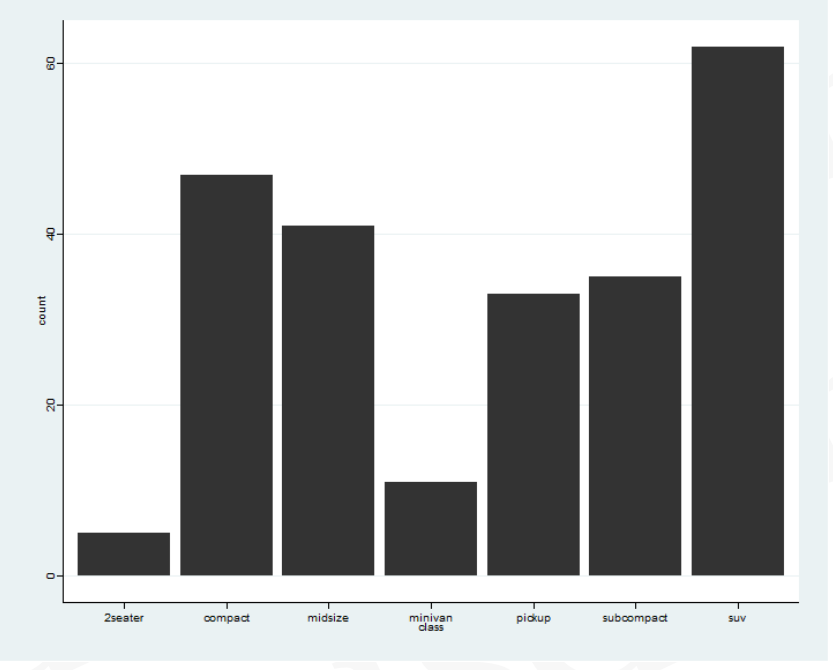
```
1    p <- ggplot(mpg, aes(x=class))
2    p + geom_bar(position='dodge') + theme_bw()
```



- ggthemes

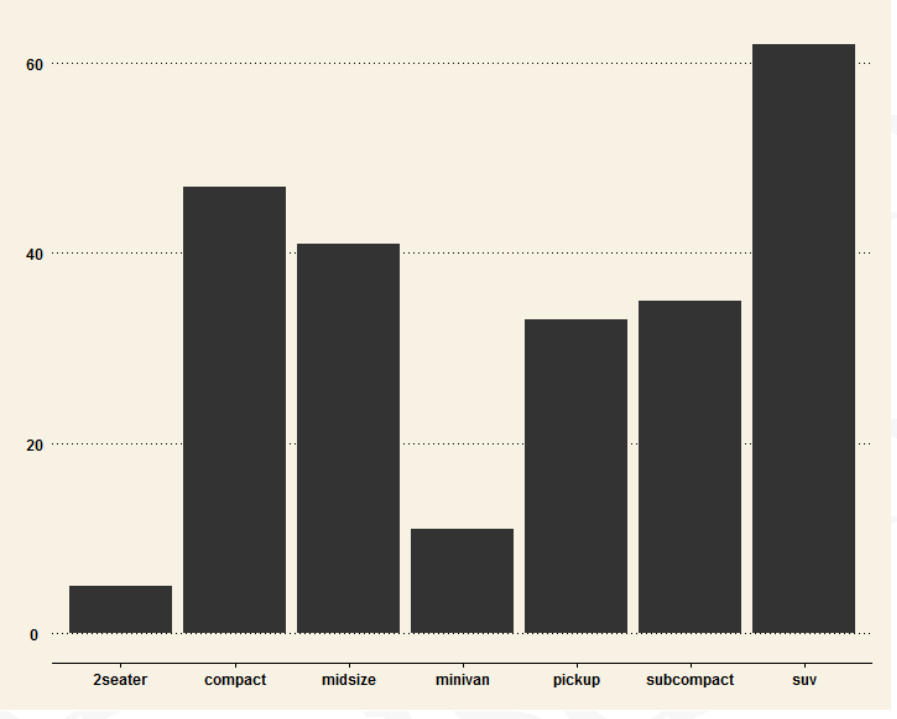
❖ stata风格

```
1    library("ggthemes")
2    p + geom_bar(position='dodge') + theme_stata()
```



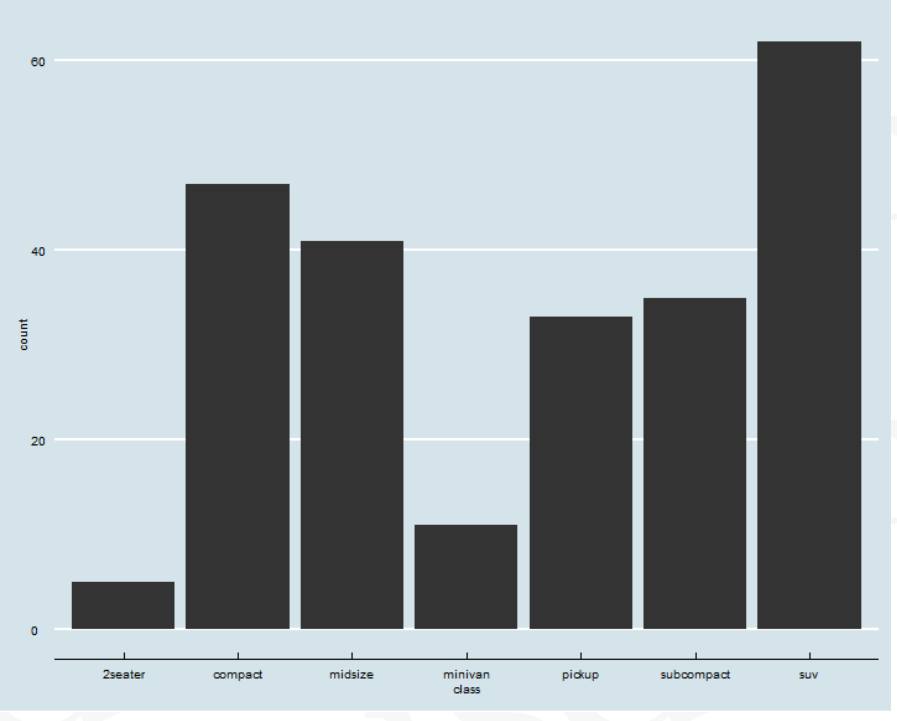
❖ 华尔街日报风格

```
1    p + geom_bar(position='dodge') + theme_wsj()
```



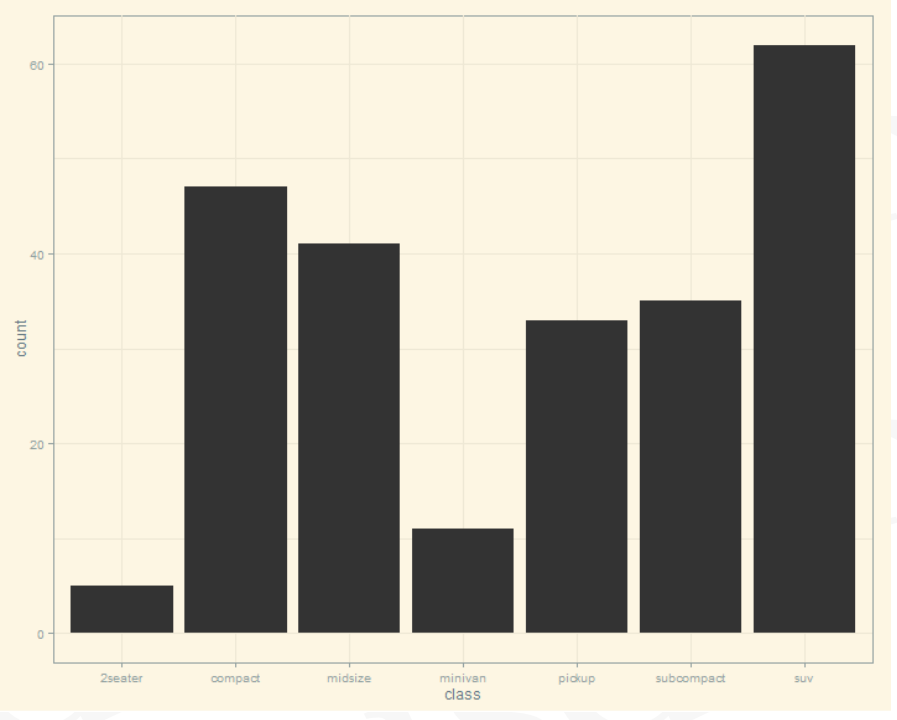
❖ 经济学人杂志风格

```
1    p + geom_bar(position='dodge') + theme_economist()
```



❖ Solarized palette

```
1    p + geom_bar(position='dodge') + theme_solarized()
```



◆ 自定义 theme()

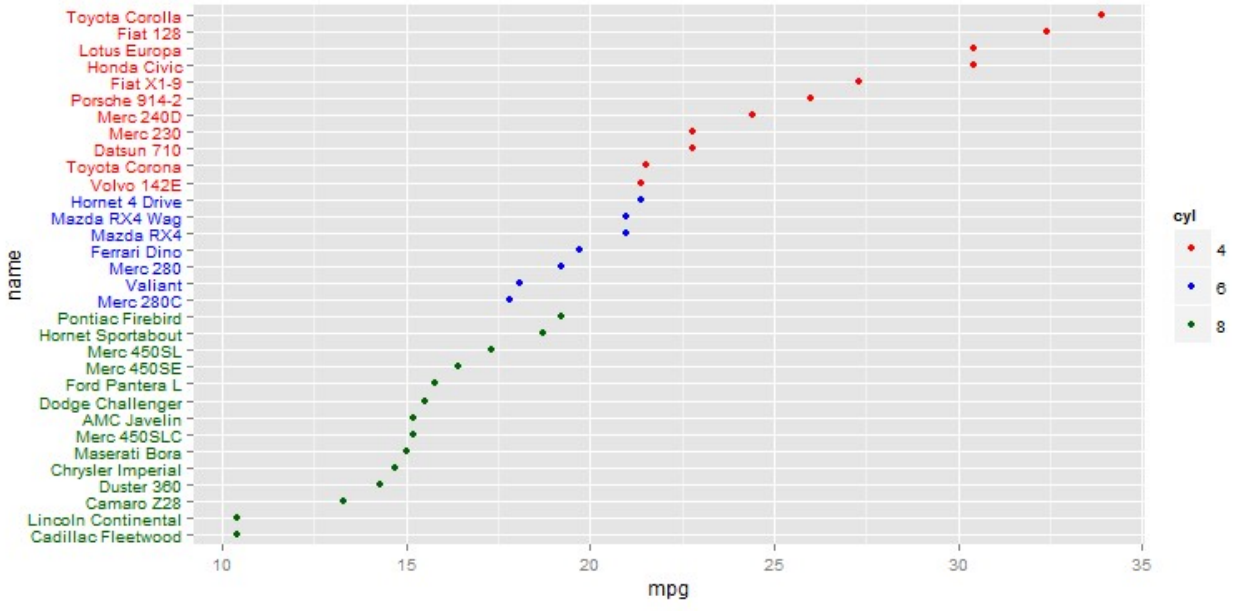
通过调用 `theme()` 可以设置图形的任意元素

实例

◆ 散点图

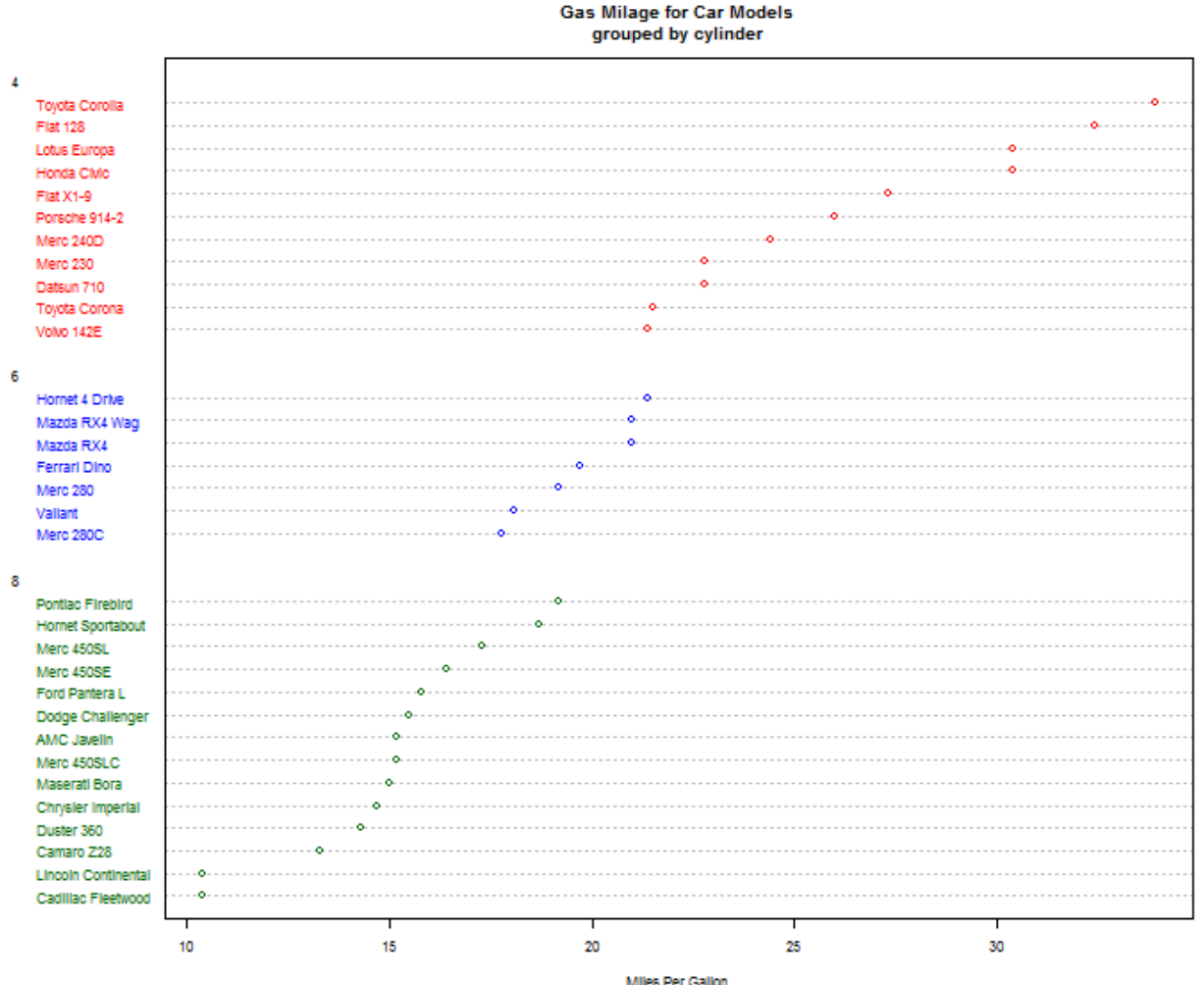
```
1   library(ggplot2)
2
3   x = mtcars
4   x$name = factor(rownames(x), levels = rownames(x))
5
6   # 散点图
7   ggplot(data = x, aes(x = name, y = mpg)    ) + geom_point(    ) +
    coord_flip()
8
9   # 针对 mpg 排序
10  x <- x[order(mtcars$mpg), ] # sort by mpg
11  x$name = factor(rownames(x), levels = rownames(x))
12  x$cyl <- factor(x$cyl) # it must be a factor
13  ggplot(data = x, aes(x = name, y = mpg, colour = cyl)) + geom_point()
    + coord_flip()
14
```

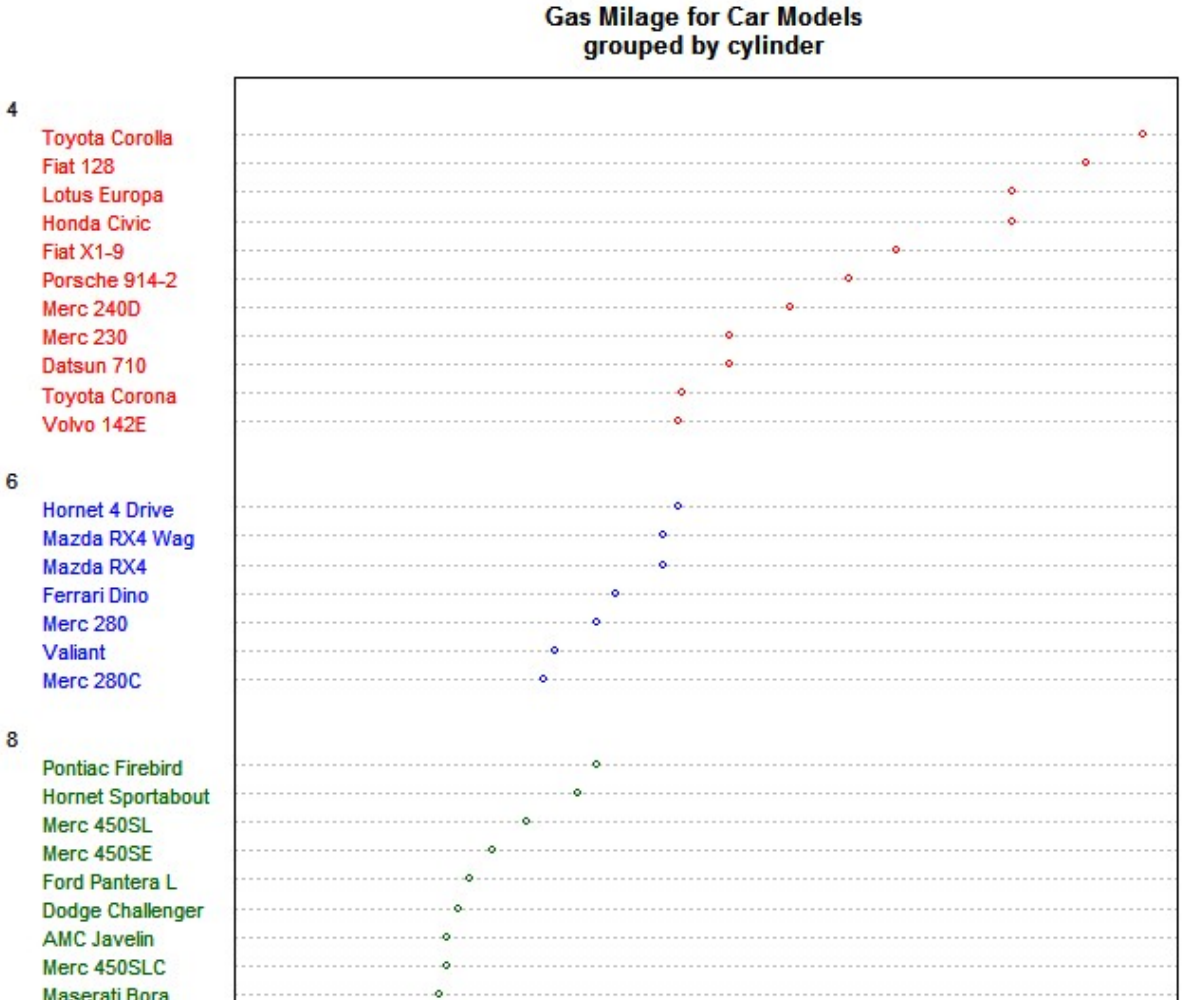
```
15  # 针对 cyl 和 mpg 排序
16  x = x[order(x$cyl, -x$mpg, decreasing = T), ]
17  x$name = factor(rownames(x), levels = rownames(x))
18  ggplot(data = x, aes(x = name, y = mpg, colour = cyl)) + geom_point()
    + coord_flip()
19
20  # Y 轴标签区分色彩
21  myPalette <- c('red', 'blue', 'darkgreen')
22  myColor = myPalette[as.factor(x$cyl)]
23  ggplot(data = x, aes(x = name, y = mpg, colour = cyl)) + geom_point()
    + coord_flip() +
24    scale_colour_manual(values = myPalette) +
25    theme(axis.text.y = element_text(colour = myColor))
```


Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1-9
Porsche 914-2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Merc 280
Valiant
Merc 280C
Pontiac Firebird
Hornet Sportabout
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
name
10
15
20
25
30
35
mpg
cyl
4
6
8

对比:

```
# Dotplot: Grouped Sorted and Colored
# Sort by mpg, group and color by cylinder
x <- mtcars[order(mtcars$mpg),] # sort by mpg
x$cyl <- factor(x$cyl) # it must be a factor
x$color[x$cyl==4] <- "red"
x$color[x$cyl==6] <- "blue"
x$color[x$cyl==8] <- "darkgreen"
dotchart(x$mpg,labels=row.names(x),cex=.7,groups= x$cyl,
  main="Gas Milage for Car Models\ngrouped by cylinder",
  xlab="Miles Per Gallon", gcolor="black", color=x$color)
```

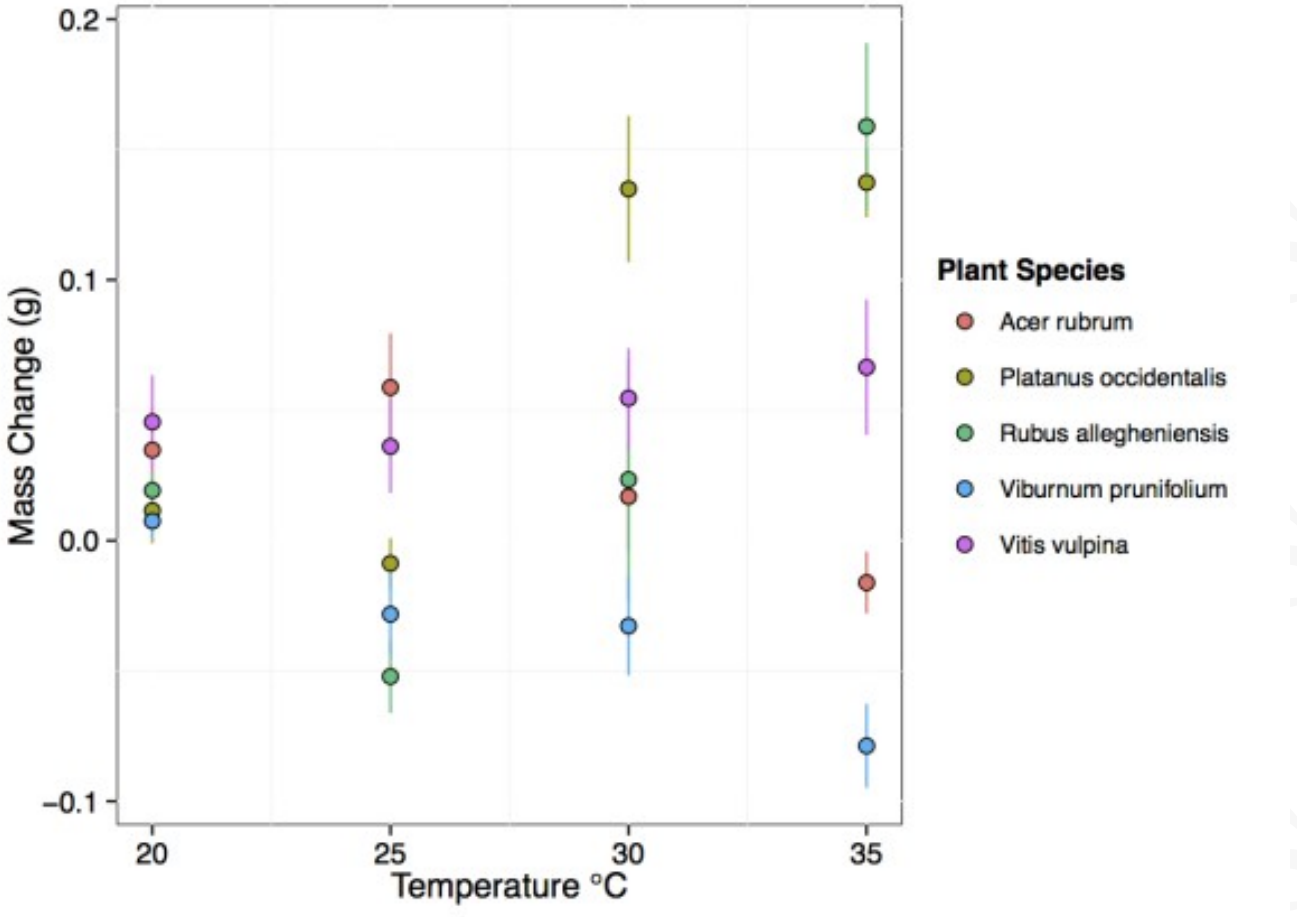

Gas Milage for Car Models
grouped by cylinder
4
Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1-9
Porsche 914-2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
6
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Merc 280
Valiant
Merc 280C
8
Pontiac Firebird
Hornet Sportabout
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
10
15
20
25
30
Miles Per Gallon


Gas Milage for Car Models
grouped by cylinder
4
Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1-9
Porsche 914-2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
6
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Merc 280
Valiant
Merc 280C
8
Pontiac Firebird
Hornet Sportabout
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora

◆ 散点误差图

```
1   # make a standard error function for plotting
2   seFunc <- function(x){
3    se <- sd(x) / sqrt(sum(!is.na(x)))
4    lims <- c(mean(x) + se, mean(x) - se)
5    names(lims) <- c('ymin', 'ymax')
6    return(lims)
7   }
8
9   # ggplot!
10  ggplot(subDat, aes(Temperature, Herb_RGR, fill = Food_Type)) +
11   stat_summary(geom = 'errorbar', fun.data = 'seFunc', width = 0,
    aes(color = Food_Type), show_guide = F) +
12   stat_summary(geom = 'point', fun.y = 'mean', size = 3, shape =
    21) +
13   ylab('Mass Change (g)') +
14   xlab(expression('Temperature '*degree*C)) +
```

```
15    scale_fill_discrete(name = 'Plant Species') +
16    theme(
17    axis.text = element_text(color = 'black', size = 12),
18    axis.title = element_text(size = 14),
19    axis.ticks = element_line(color = 'black'),
20    legend.key = element_blank(),
21    legend.title = element_text(size = 12),
22    panel.background = element_rect(color = 'black', fill = NA)
23    )
```

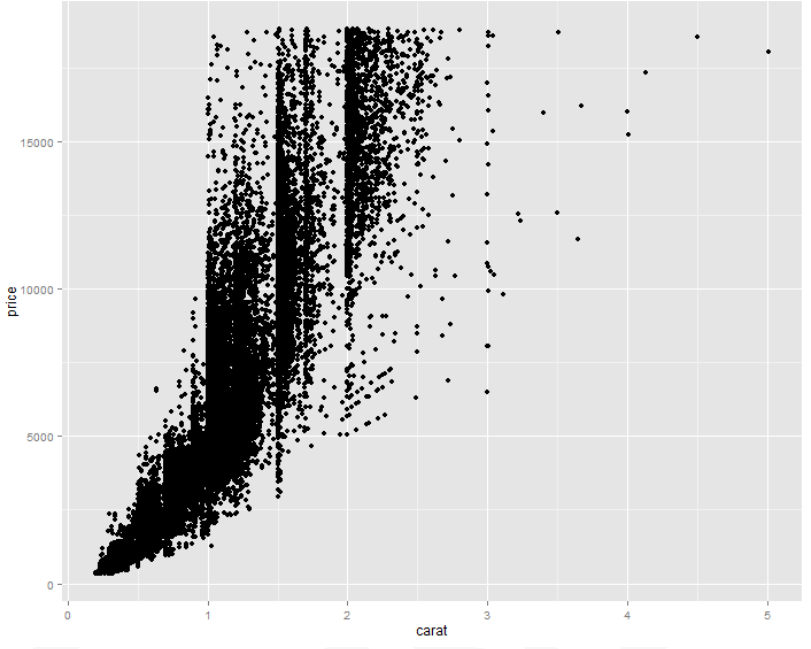

0.2
0.1
0.0
−0.1
Mass Change (g)
20
25
30
35
Temperature °C
Plant Species
Acer rubrum
Platanus occidentalis
Rubus allegheniensis
Viburnum prunifolium
Vitis vulpina

观察密集散点的方法

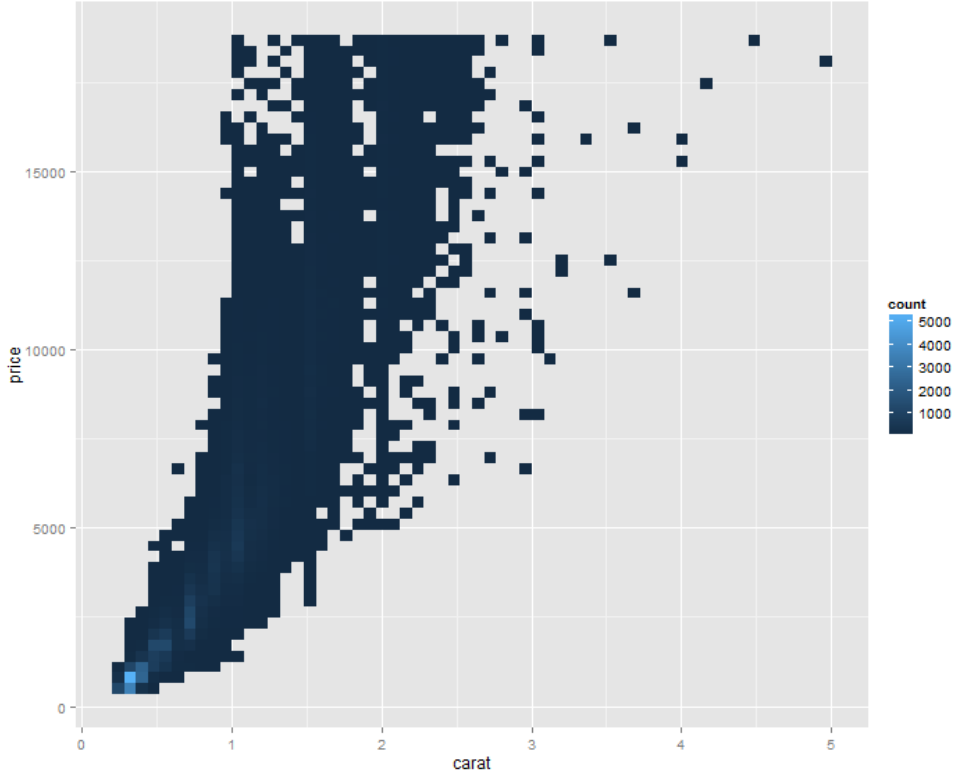
- 增加扰动（jitter）
- 增加透明度（alpha）
- 二维直方图（stat_bin2d）
- 密度图（stat_density2d）

◆ 示例

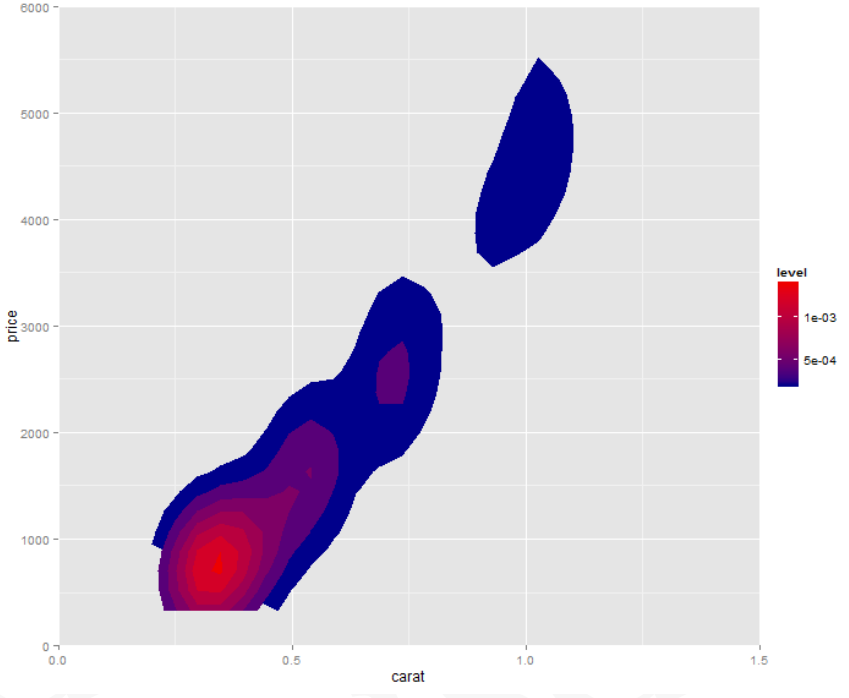
```
1    p <- ggplot(diamonds,aes(carat,price))
2    p + geom_point()
```



```
1    p + stat_bin2d(bins = 60)
```

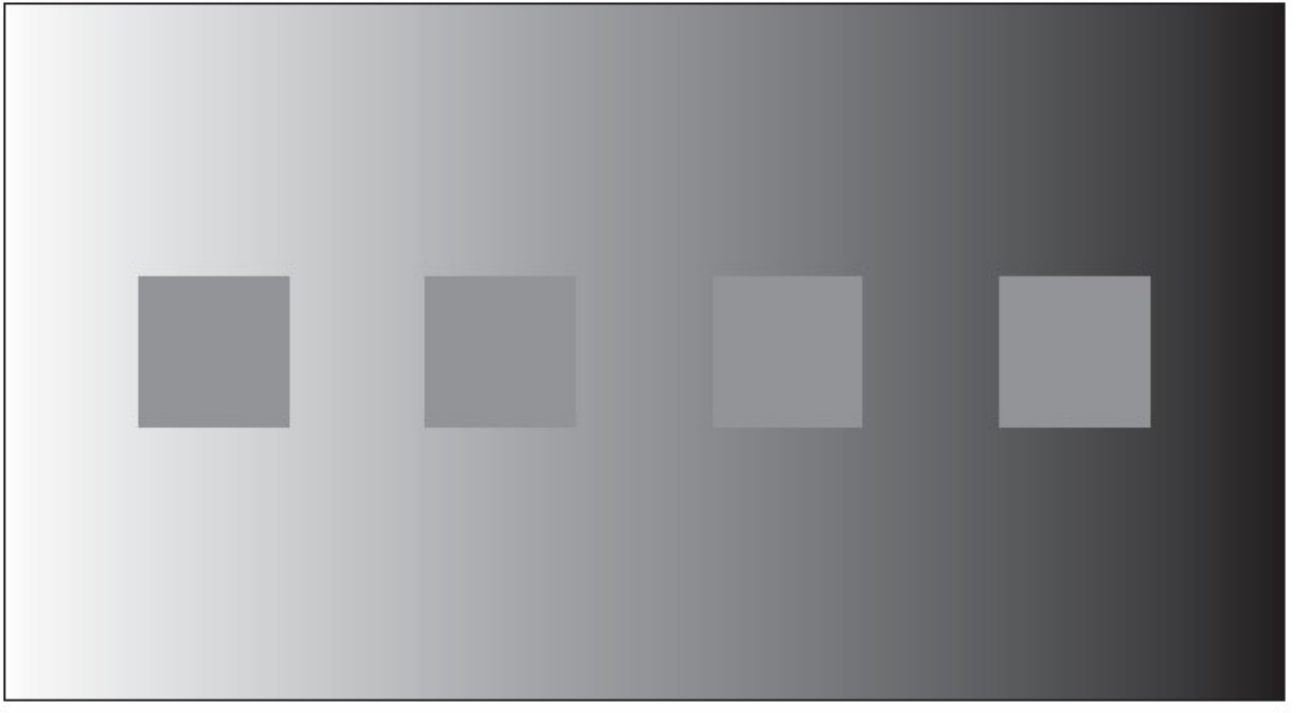


```
1    p + stat_density2d(aes(fill = ..level..), geom="polygon") +
2      coord_cartesian(xlim = c(0, 1.5),ylim=c(0,6000))+
3      scale_fill_continuous(high='red2',low='blue4')
```



如何使用颜色

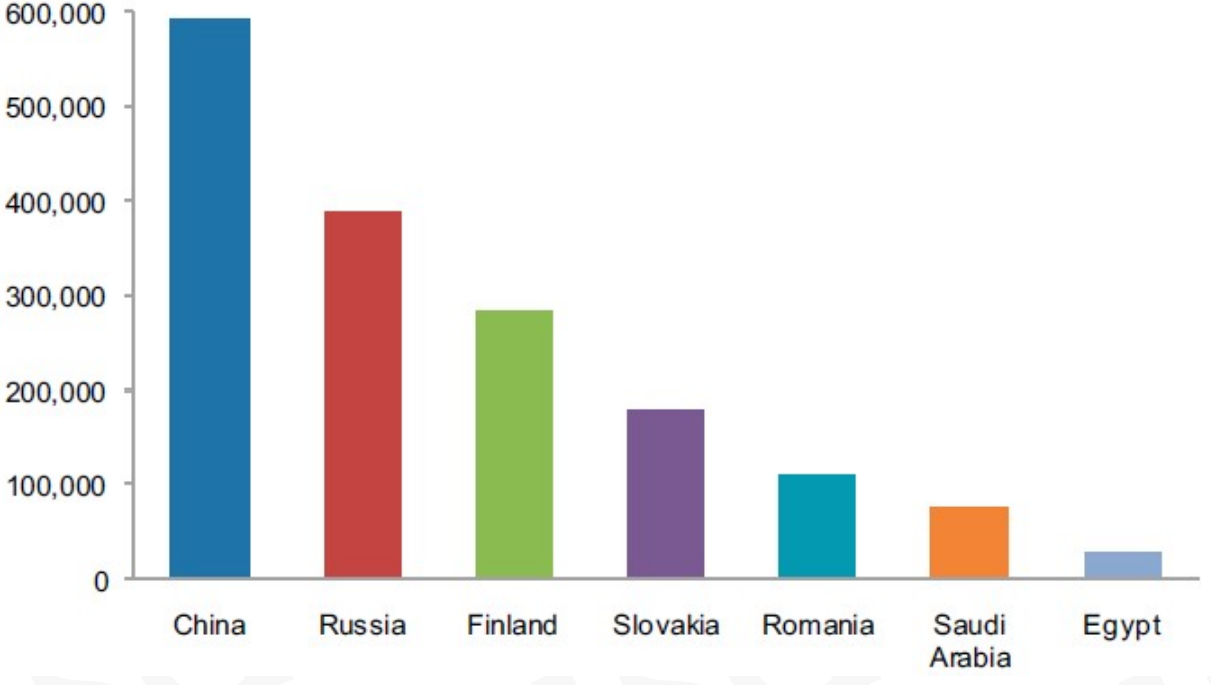
◆ 背景色

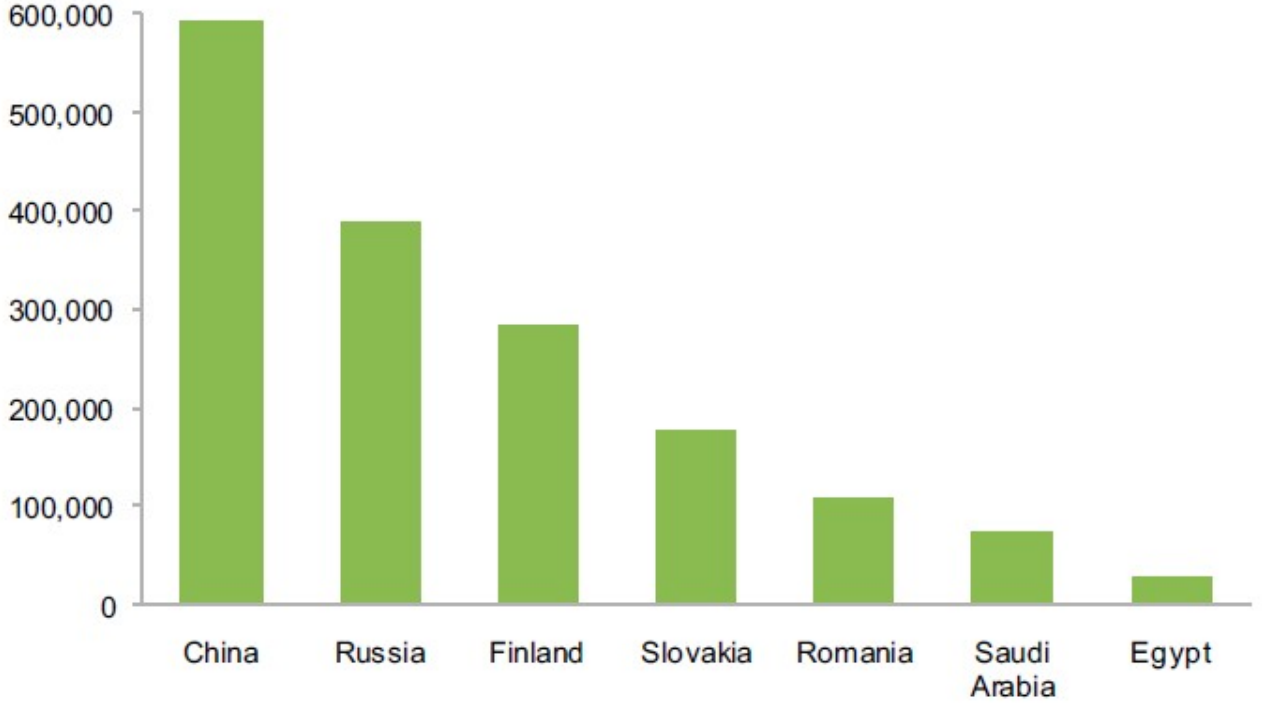


- 确保图表中的背景颜色一致，避免使用渐变色；
- 为了突出图表中的信息，背景使用有效的对比色；

◆ 颜色的意义

- 颜色必须服务于特定的信息传递目的;




600,000
500,000
400,000
300,000
200,000
100,000
0
China
Russia
Finland
Slovakia
Romania
Saudi Arabia
Egypt

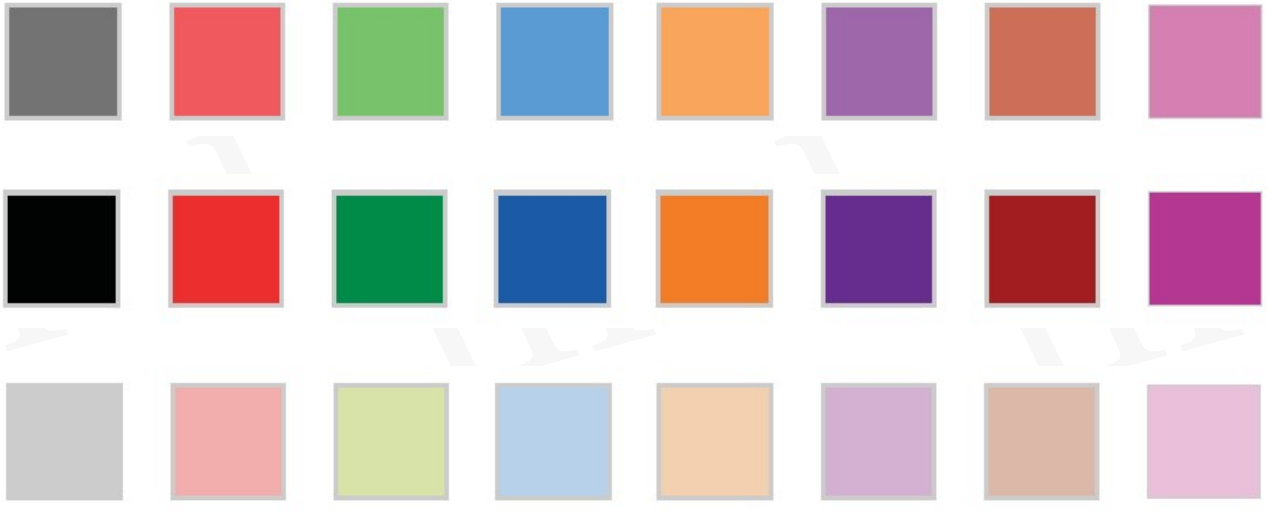
◆ 调色板

为不同目的，定义不同的调色板：

- 不是所有的信息都是相等的；用颜色来强调需要传达的信息；
- 高亮特殊数据：用颜色加深或提亮重点数据；
- 组合元素：用颜色统一同组元素，区分它组元素；
- 对数值进行编码；

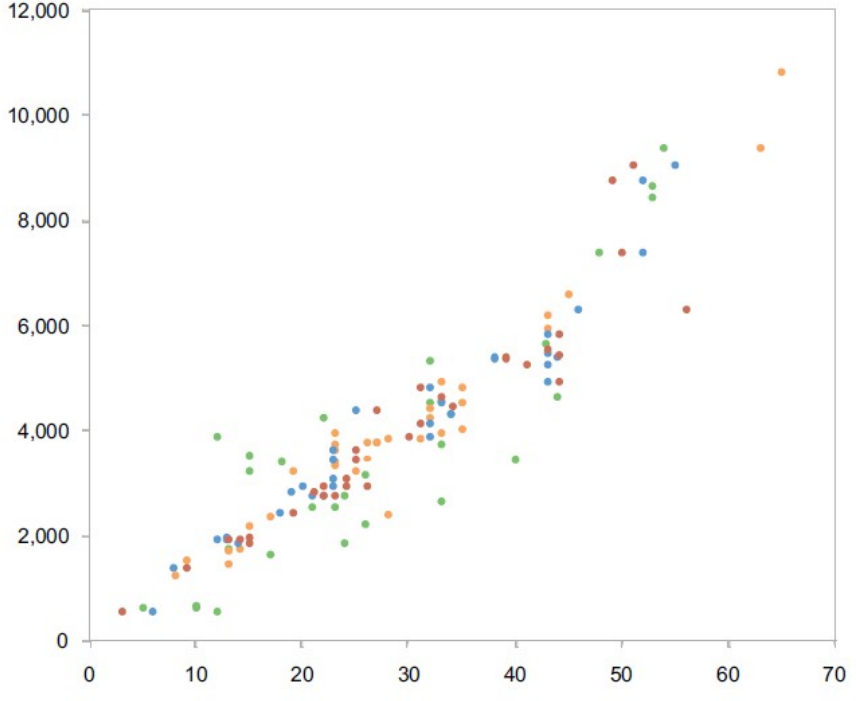
用色原则：

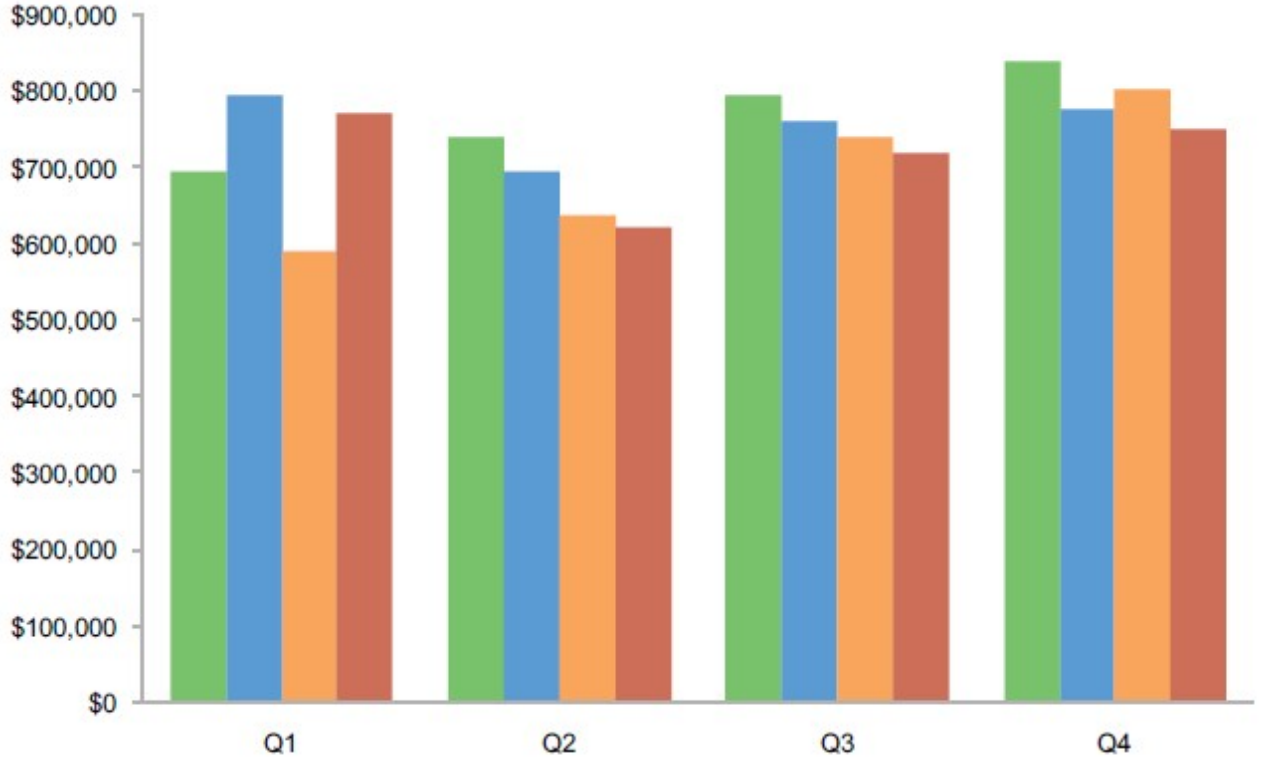
- 使用柔和的、自然的颜色显示绝大部分信息；使用明亮/暗的颜色来突出显示；
- 使用在色谱上等距的不同色调，以避免强度差异；



❖ 颜色承载对象的大小、粗细

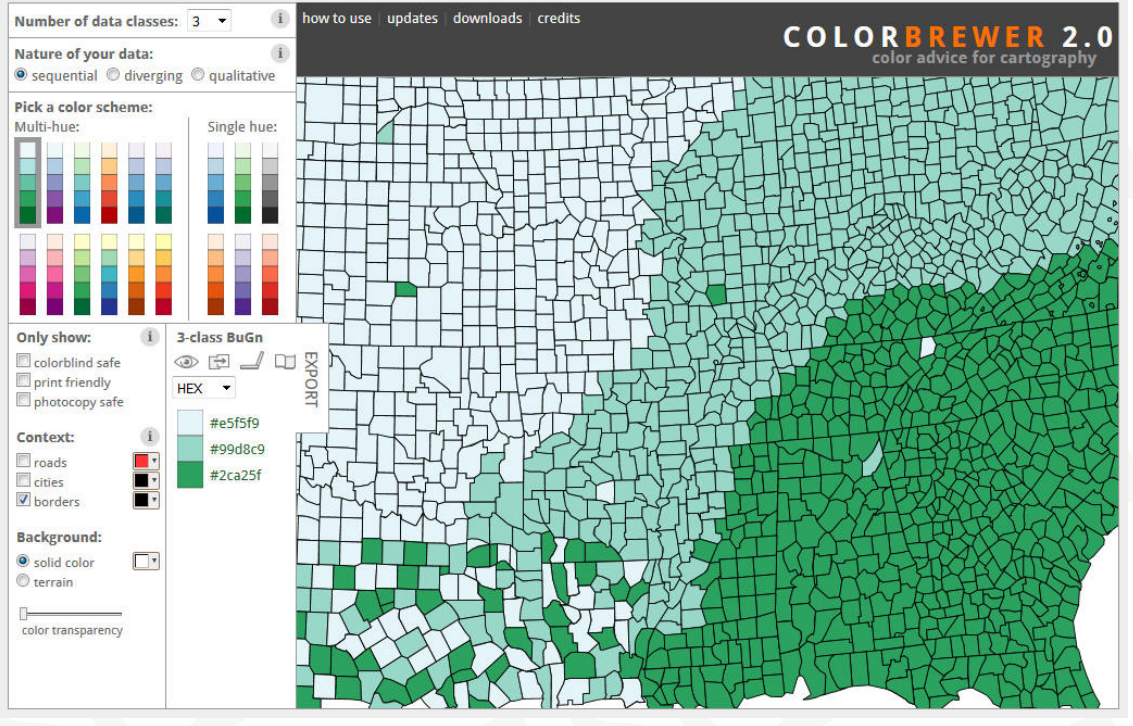
当承载颜色的对象面积过小、线条过细时，颜色难以分辨




$900,000
$800,000
$700,000
$600,000
$500,000
$400,000
$300,000
$200,000
$100,000
$0
Q1
Q2
Q3
Q4

❖ 调色板参考网站

`http://colorbrewer2.org/`



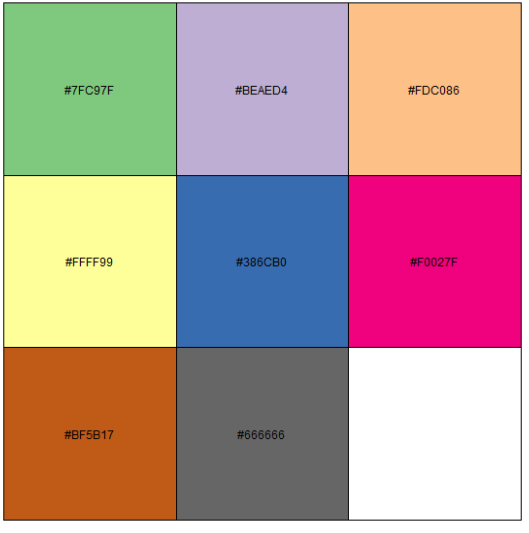
❖ 调用R的调色板

R 包 `scales` 提供了调色板调用函数

```
brewer_pal(type = "seq", palette = 1, direction = 1)
```

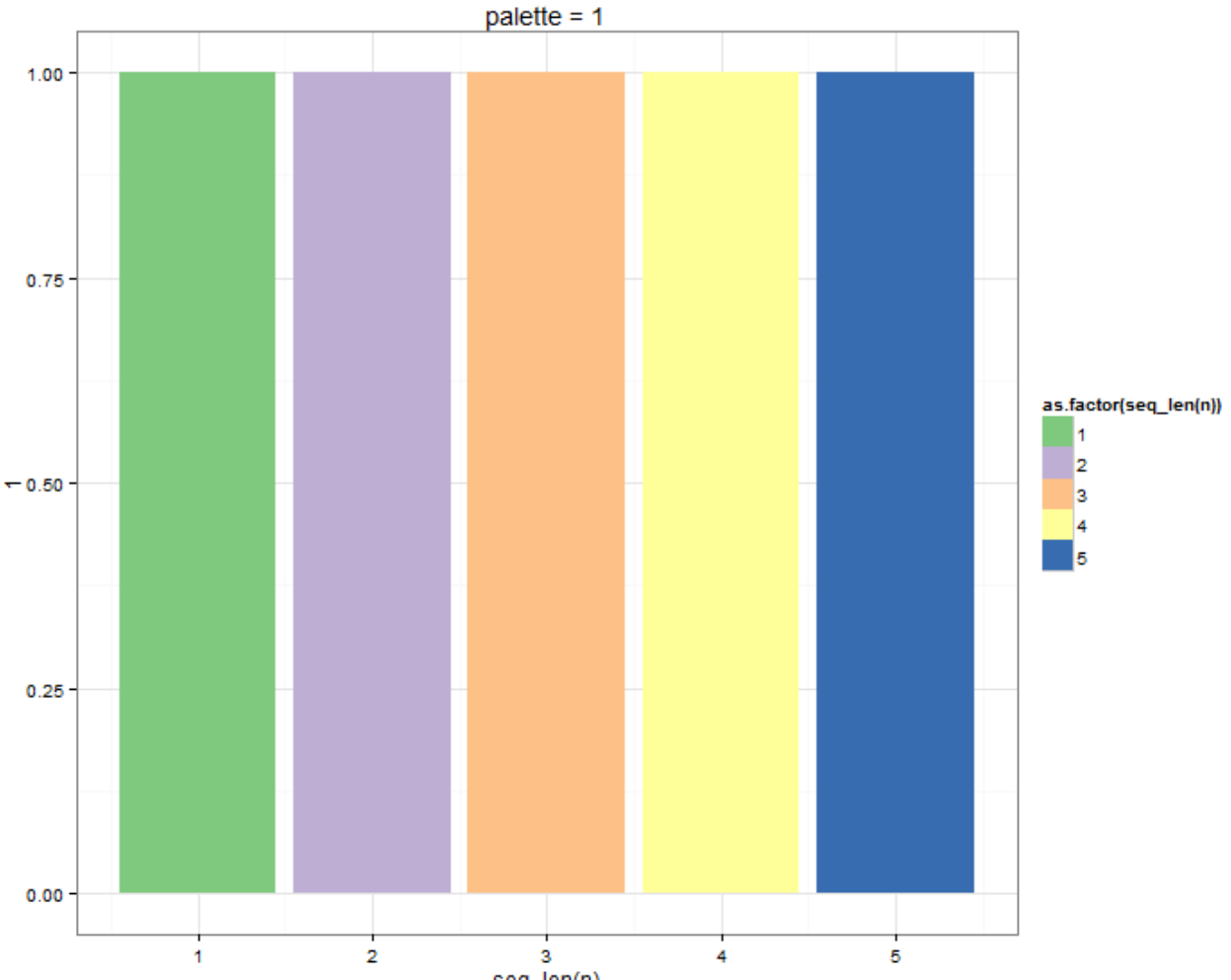
- `type` seq (sequential), div (diverging) or qual (qualitative)
- `palette` 调色板顺序数值，1-n不等

```
library(scales)
for (i in seq_len(1)) {
  cat(paste('quallitative', i))
  show_col(brewer_pal(type='qual', pal = i)(9))
}
## quallitative 1
```



❖ 调色板与 ggplot2

```
1  library(ggplot2)
2  n=5
3  ggplot(data=NULL, aes(x=seq_len(n), y=1, fill = as.factor(seq_len(n))))
   +
4      geom_bar(stat ='identity') +
5      scale_fill_brewer(type = 'qual', palette = 1) +
6      theme_bw() + ggtitle(paste('palette =', 1))
```


palette = 1
1.00
0.75
0.50
0.25
0.00
1
1
2
3
4
5
seq_len(n)
as.factor(seq_len(n))
1
2
3
4
5