

ggplot2 简介

周世祥

2015 年 10 月 29 日

ggplot2 是做什么的

每当我们看到一个新的软件，第一反应会是：为什么又要发明一个新软件？ggplot2 是 R 世界里相对还比较年轻的一个包，在它之前，官方 R 已经有自己的基础图形系统（graphics 包）和网格图形系统（grid 包），并且 Deepayan Sarkar 也开发了 lattice 包，看起来 R 的世界对图形的支持已经足够强大了。那么我们不禁要问，为什么还要发明一套新的系统？

设计理念

打个比方，想想我们小时候怎样学中文的。最开始的时候我们要识字，不认识字就没法阅读和写作，但我们并不是一直按照一个个汉字学习的，而是通过句子和具体的场景故事学习的。为什么不在小学时背六年字典呢？那样可能认识所有的汉字。原因很简单，光有单字，我们不会说话，也无法阅读和写作。缺的是什么？答案是对文字的组织能力，或者说语法。

R 的基础图形系统基本上是一个“纸笔模型”，即：一块画布摆在面前，你可以在这里画几个点，在那里画几条线，指哪儿画哪儿。后来 lattice 包的出现稍微改善了这种情况，你可以说，我要画散点图或直方图，并且按照某个分类变量给图中的元素上色，此时数据才在画图中扮演了一定的中心角色，我们不用去想具体这个点要用什么颜色（颜色会根据变量自动生成）。然而，lattice 继承了 R 语言的一个糟糕特征，就是参数设置铺天盖地，足以让人窒息，光是一份 xyplot() 函数的帮助文档，恐怕就够我们消磨一天时间了，更重要的是，lattice 仍然面向特定的统计图形，像基础图形系统一样，有直方图、箱线图、条形图等等，它没有一套可以让数据分析者说话的语法。

那么数据分析者是怎样说话的呢？他们从来不会说这条线用 #FE09BE 颜色，那个点用三角形状，他们只会说，把图中的线用数据中的职业类型变量上色，或图中点的形状对应性别变量。有时候他们画了一幅散点图，但马上他们发现这幅图太拥挤，最好是能具体看一下里面不同收入阶层的特征，所以他们会说，把这幅图拆成七幅小图，每幅图对应一个收入阶层。然后发现散点图的趋势不明显，最好加上回归直线，看看回归模型反映的趋势是什么，或者发现图中离群点太多，最好做一下对数变换，减少大数值对图形的主导性。

从始至终，数据分析者都在数据层面上思考问题，而不是拿着水彩笔和调色板在那里一笔一划作图，而计算机程序员则倾向于画点画线。Leland Wilkinson 的著作在理论上改善了这种状况，他提出了一套图形语法，让我们在考虑如何构建一幅图形的时候不再陷在具体的图形元素里面，而是把图形拆分为一些互相独立并且可以自由组合的成分。这套语法提出来之后他自己也做了一套软件，但显然这套软件没有被广泛采用；幸运的是，Hadley Wickham 在 R 语言中把这套想法巧妙地实现了。

为了说明这种语法的想法，我们考虑图形中的一个成分：坐标系。常见的坐标系有两种：笛卡尔坐标系和极坐标系。在语法中，它们属于一个成分，可自由拆卸替换。笛卡尔坐标系下的条形图实际上可以对应极坐标系下的饼图，因为条形图的高可以对应饼图的角度，本质上没什么区别。因此在 ggplot2 中，从一幅条形图过渡到饼图，只需要加极少量的代码，把坐标系换一下就可以了。如果我们用纸笔模型，则可以想象，这完全是不同的两幅图，一幅图里面要画的是矩形，另一幅图要画扇形。

更多的细节在本书中会介绍，这里我们只是简略说明用语法画图对用纸笔画图来说在思维上的优越性；前者是说话，后者是说字。

发展历程

ggplot2 是 Hadley 在爱荷华州立大学博士期间的作品，也是他博士论文的主题之一，实际上 ggplot2 还有个前身 ggplot，但后来废弃了，某种程度上这也是 Hadley 写软件的特征，熟悉他的人就知道这不是他第一个“2”版本的包了（还有 reshape2）。带 2 的包和原来的包在语法上会有很大的改动，基本上不兼容。尽管如此，他的

R 代码风格在 R 社区可谓独树一帜，尤其是他的代码结构很好，可读性很高，ggplot2 是 R 代码抽象的一个杰作。读者若感兴趣，可以在 GitHub 网站上浏览他的包：。在用法方面，ggplot2 也开创了一种奇特而绝妙的语法，那就是加号：一幅图形从背后的设计来说，是若干图形语法的叠加，从外在的代码来看，也是若干 R 对象的相加。这一点精妙尽管只是 ggplot2 系统的很小一部分，但我个人认为没有任何程序语言可比拟，它对作为泛型函数的加号的扩展只能用两个字形容：绝了。

至 2013 年 2 月 26 日，ggplot2 的邮件列表 (<http://groups.google.com/group/ggplot2>) 订阅成员已达 3394 人，邮件总数为 15185 封，已经成为一个丰富、活跃的用户社区。未来 ggplot2 的发展也将越来越依赖于用户的贡献，这也是很多开源软件最终的走向。

欢迎来到 **ggplot2** 的世界

ggplot2 是一个用来绘制统计图形的 R 软件包。由一套图形语法所支持。由一系列的独立图形部件组成，并能以多种不同方式组合起来，这一点使得 ggplot2 的功能非常强大，因为她不会局限于一些已经定义好的统计图形，而是根据你的需要量身定做。

ggplot2 采用图层的设计方式，可以从原始图层开始，首先绘制原始数据，然后不断添加图形注释和统计汇总结果。这种绘图方式与分析问题中的结构化的思维是一致的。能缩短你的“所思”与“所见”的距离。

在线访问网站或 R 帮助系统获得 ggplot2 内置文档。一些数据集见：[\(http://ggplot2.org/book/\)](http://ggplot2.org/book/). 帮助文档：[\(http://docs.ggplot2.org/current/\)](http://docs.ggplot2.org/current/).

什么是图形的语法

Wilkinson 于 2005 年创建了一套用来描述所有统计图形深层特性的语法规则，该语法回答了“什么是统计图形”这一问题。一张统计图形就是从数据到几何对象 geom 的图形属性 aes 的一个映射。图形中可能还包含数据的统计变换 stats，最后绘制在特定的坐标系 coord，而分面 facet 则可以用来生成数据的不同子集的图形。一张统计图形就是由上述这些独立的图形部件所组成。ggplot2 是用于绘图的 R 语言扩展包，其理念根植于《Grammar of Graphics》一书。它将绘图视为一种映射，即从数学空间映射到图形元素空间。例如将不同的数值映射到不同的色彩或透明度。该绘图包的特点在于并不去定义具体的图形（如直方图，散点图），而是定义各种底层组件（如线条、方块）来合成复杂的图形，这使它能以非常简洁的函数构建各类图形，而且默认条件下的绘图品质就能达到出版要求。

几何对象：点线面，多边形等。上面指定的图形属性需要呈现在一定的几何对象上才能被我们看到，这些承载图形属性的对象可能是点，可能是线，可能是 bar。

标度：scale 作用是将数据映射到图形空间，如用颜色，大小或形状来表示不同的数值。

安装

确保是较新版本的 R，可从 R 综合典藏网上下载。执行 `install.package("ggplot2")`。

从 **qplot** 开始入门

概述

qplot 意思是快速作图，与 plot 很像，可以通过 R 的帮助系统，`?qplot` 来获取该函数的帮助信息。

我们选取 diamonds 数据集做测试，此数据集包含了约 54000 颗钻石的价格和质量，数据正好在 ggplot 包中，涵盖了反映钻石质量的四个 C，克拉重量 (carat)，切工 (cut)，颜色 (color) 和净度 (clarity)，以及 5 个物理指标，深度 (depth)，钻石宽度 (table)，x，y 和 z。

我们同时使用另一个数据集，dsmall，是原始数据的一个容量为 100 的随机样本，我们用这个小样本的数据作图。

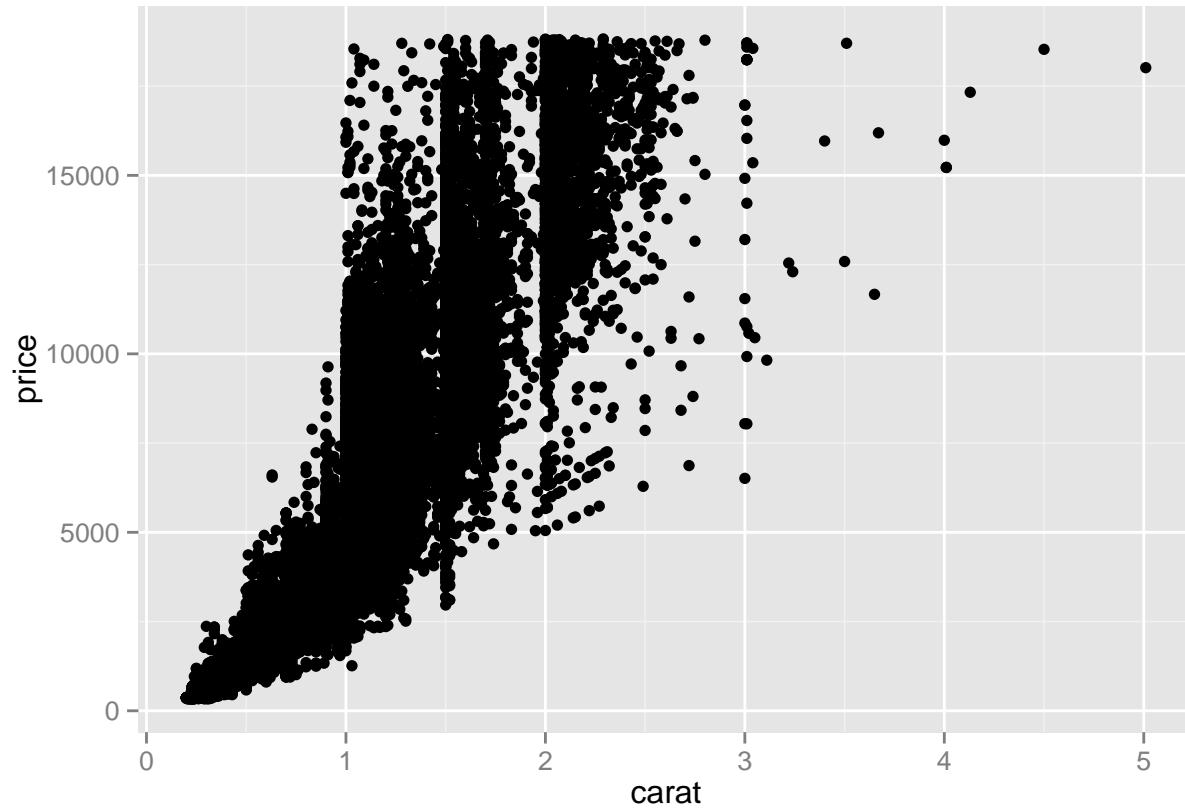
```
library(ggplot2)

set.seed(1410) # 让样本可重复

dsmall <- diamonds[sample(nrow(diamonds), 100), ]
```

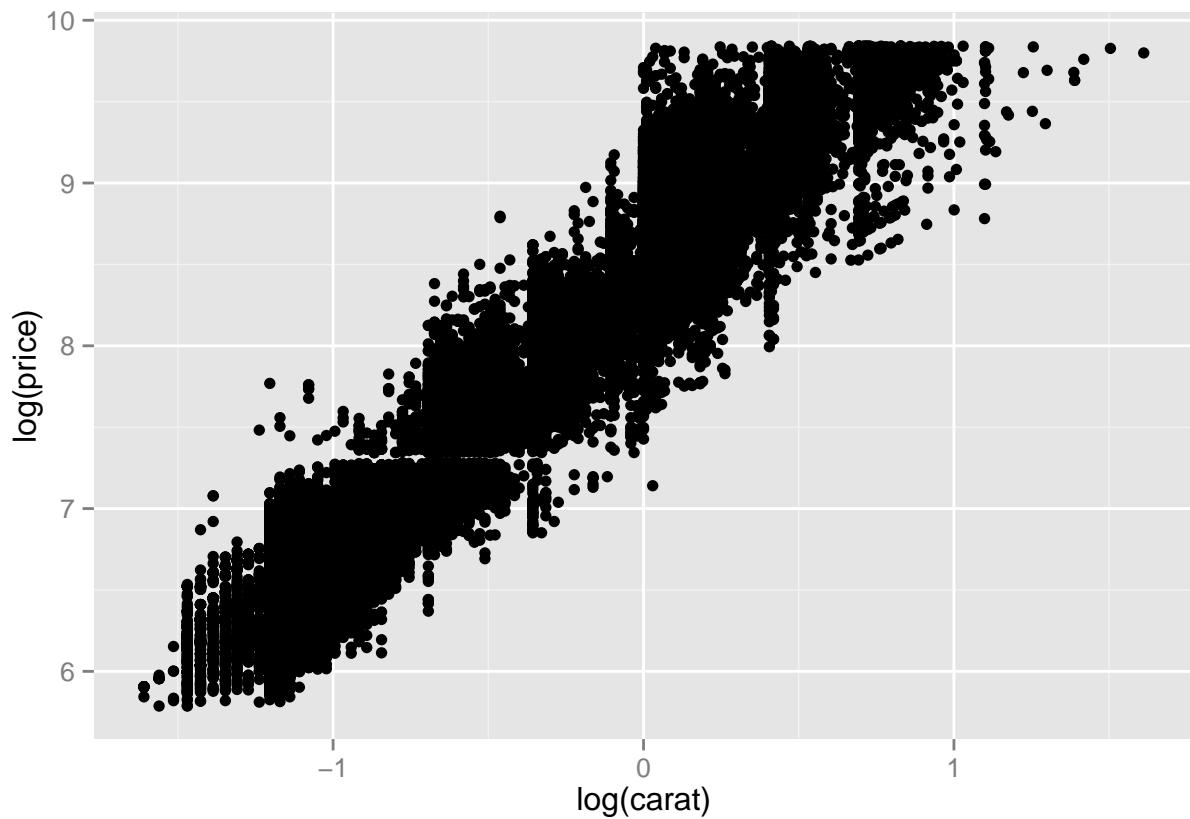
与 plot 类似，qplot 前两个参数是 x 和 y，分别代表所画图形的横坐标和纵坐标。

```
qplot(carat, price, data = diamonds)
```



这张图显示了变量之间很强的相关关系，以及一些明显的异常值，此外，在竖直方向有一些有趣的条纹，这种相关似乎是指数型的，我们应该先对变量进行一些变换。

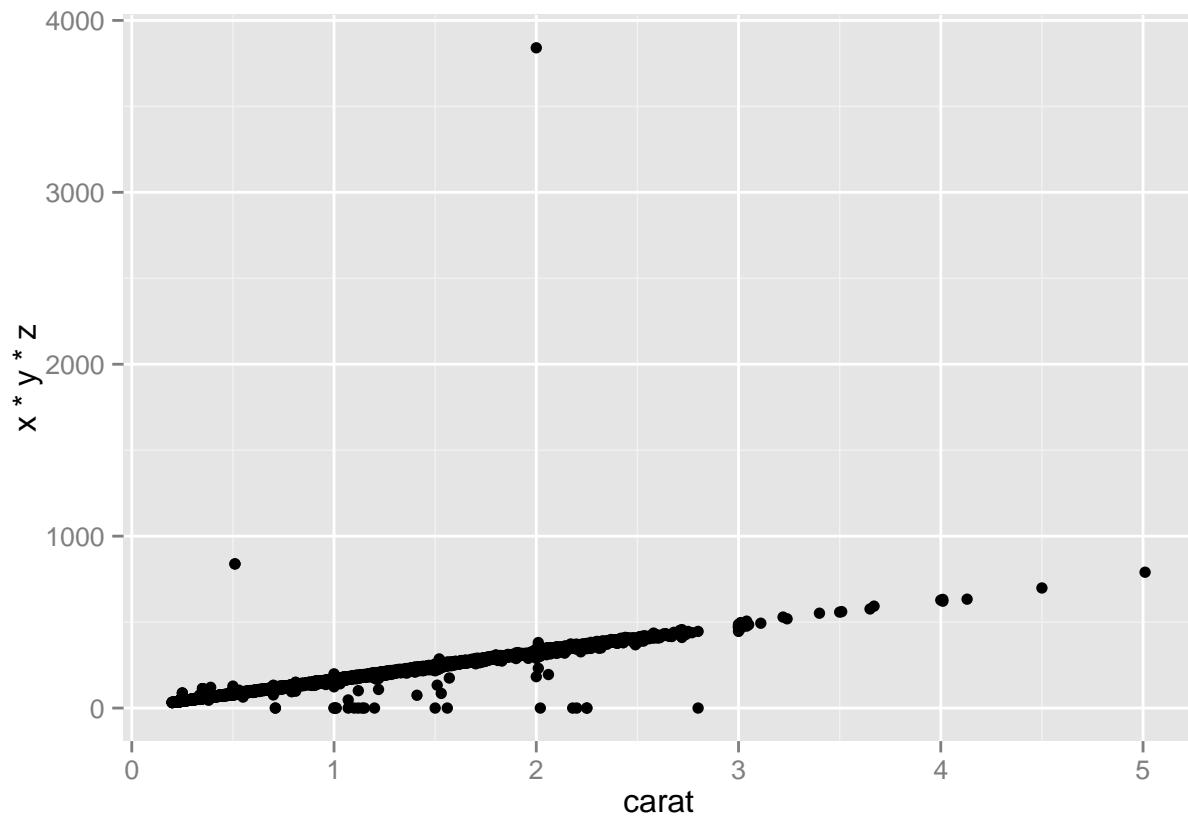
```
qplot(log(carat), log(price), data = diamonds)
```



现在这种关系就接近于线性了，然而，由于图中元素有很大重叠，所以下结论不能太轻率。

函数的参数同样可以是已有变量的某种组合。若对钻石的体积和其重量之间的关系感兴趣，可以这样做。

```
qplot(carat, x * y * z, data = diamonds)
```

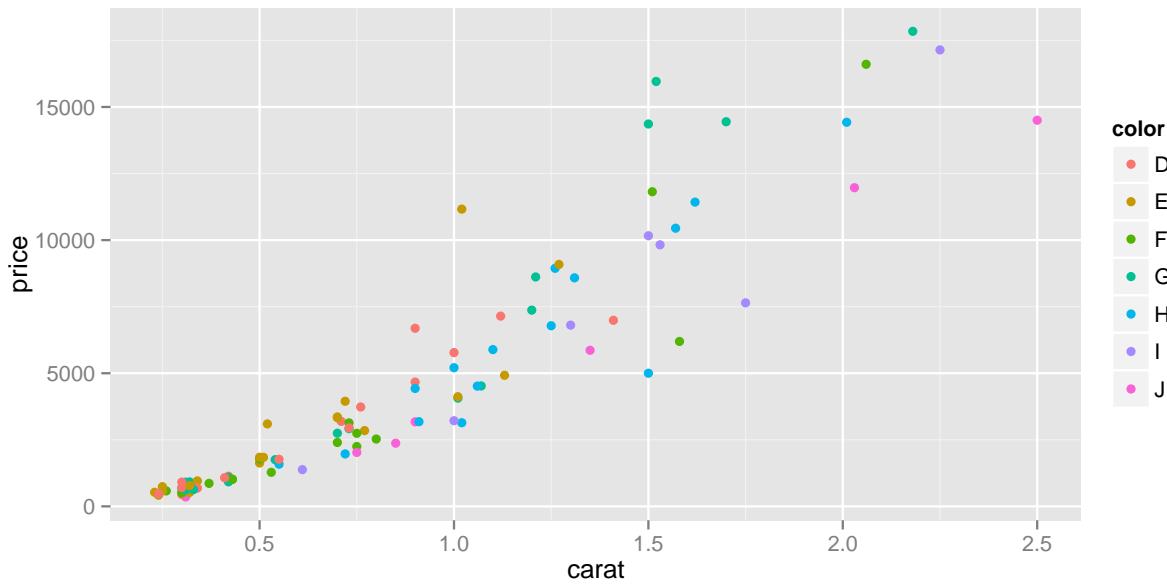


可以预料的是钻石的密度是个常数。大部分钻石都落在同一条直线上，但依然有一些大的异常点。

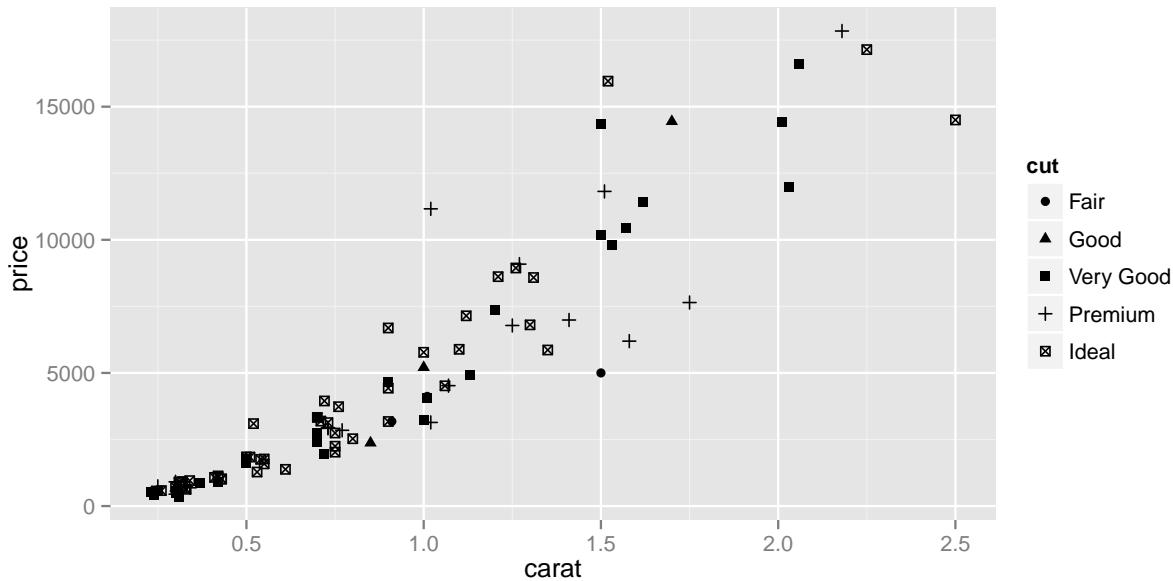
设定颜色，大小，形状和其他图形属性

与 plot 最大的不同是给图中的点设定颜色时采用不同的实现方式。下面的例子，给重量和价格的散点图添加了颜色和切工的信息。

```
# 将 color 变量映射到点的颜色 , cut 变量映射到点的形状
par(mfrow=c(1, 2))
qplot(carat, price, data = dsmall, colour = color)
```



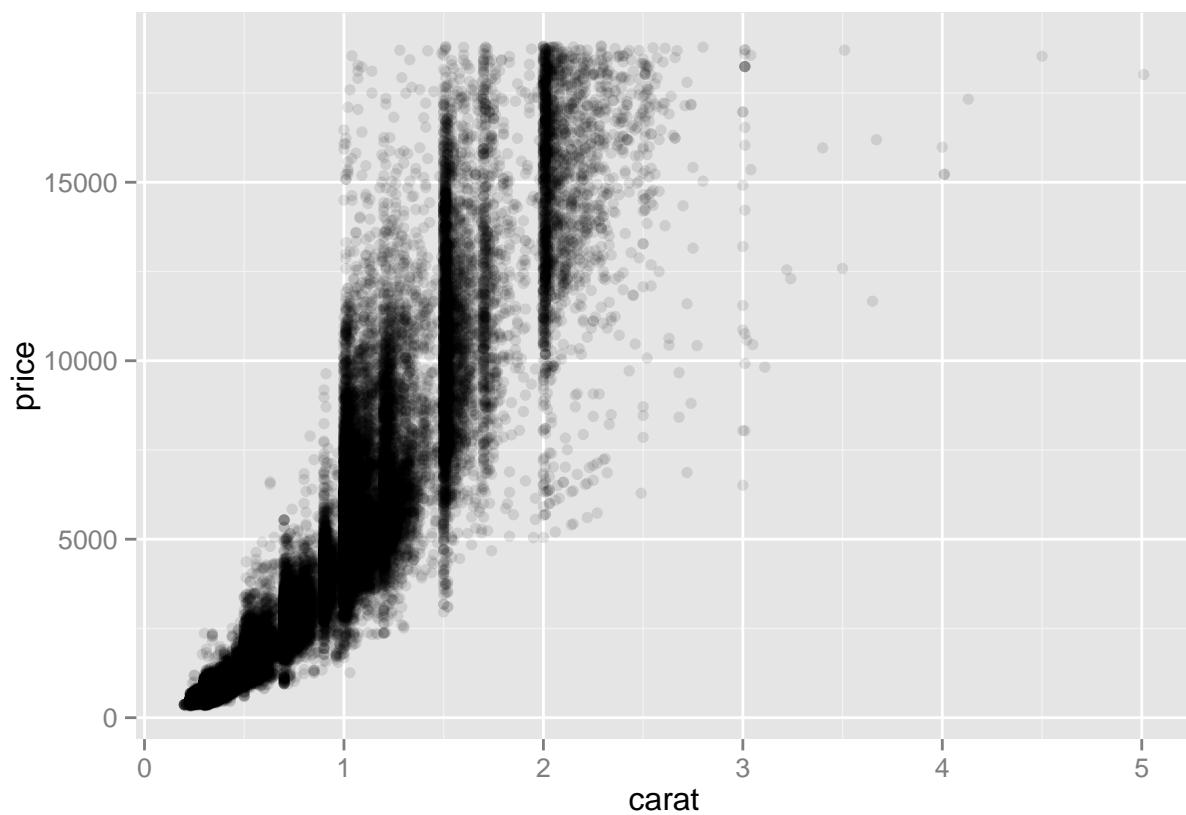
```
qplot(carat, price, data = dsmall, shape = cut)
```



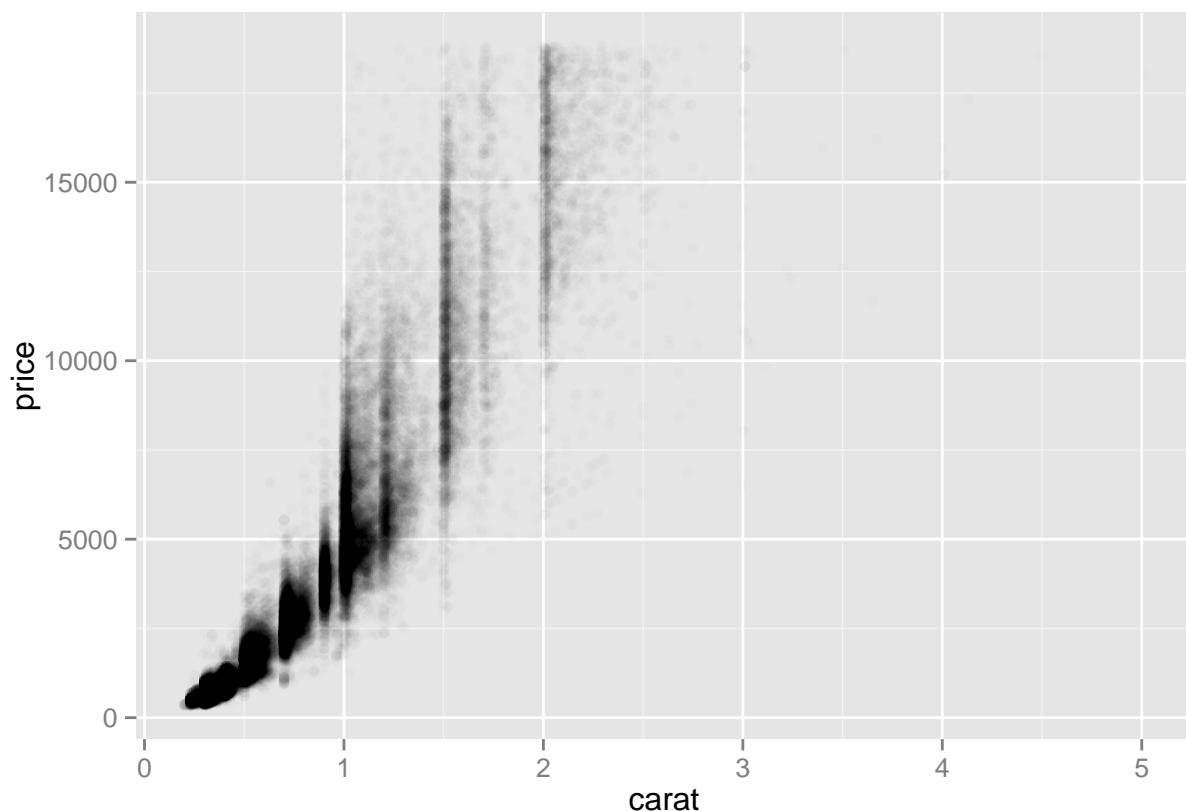
颜色，大小和形状都是图形属性的具体例子，影响数据如何进行展示的视觉属性。每一个图形属性都对应一个称为标度的函数，作用是将数据的取值映射到该图形属性的有效取值，颜色标度将J映射为紫色，将F映射为绿色。可以用手动设置图形属性，`colour=I("red")`。

对于大数据而言，使用半透明的颜色，透明度用分数来表示，`1/10`，其分母表示经过多少次重叠之后颜色将变得不透明。

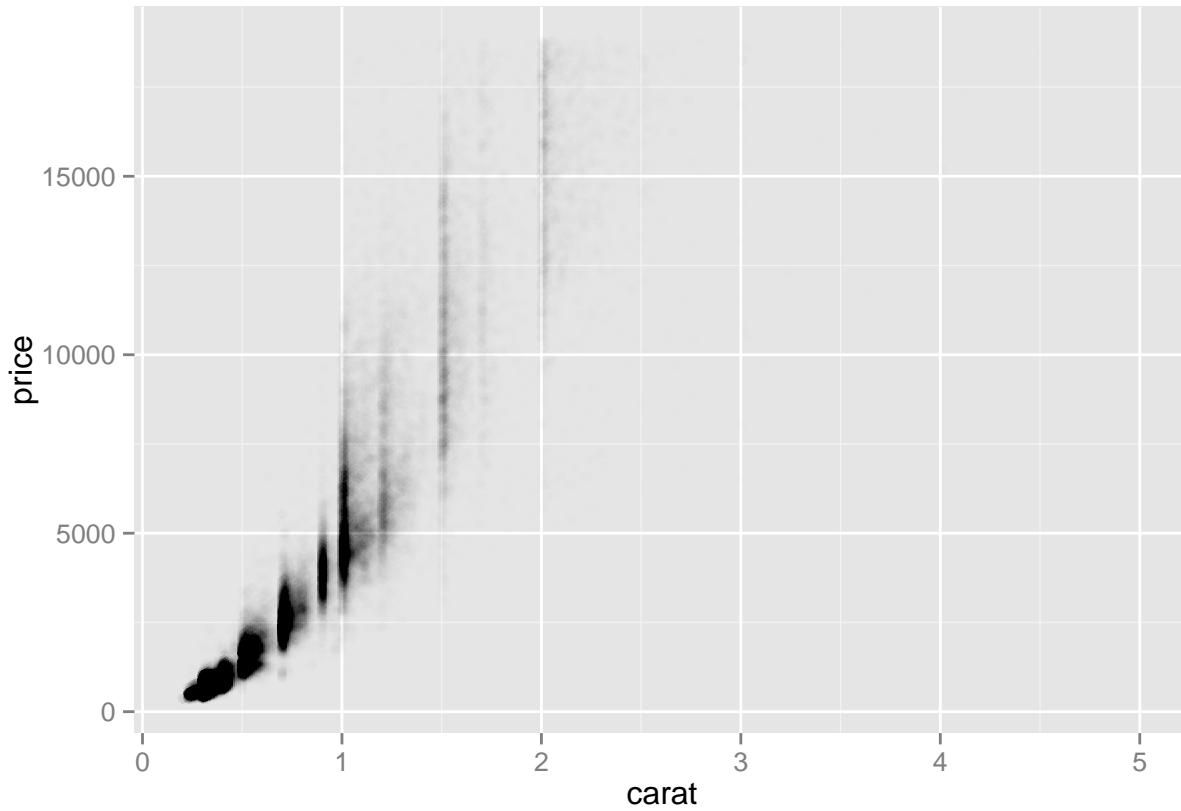
```
## 将 alpha 值从 1/10(左) 变动到 1/100(中) 再到
## 1/200(右)，来看大部分的点在哪里 进行重叠。
par(mfrow=c(1,3))
qplot(carat, price, data = diamonds, alpha = I(1/10))
```



```
qplot(carat, price, data = diamonds, alpha = I(1/100))
```



```
qplot(carat, price, data = diamonds, alpha = I(1/200))
```



不同类型的变量有不同适用的图形属性。颜色和形状适合于分类变量，而大小适合于连续变量。数据量的大小同样也影响，若数据太大，那么不同组的数据之间就很难进行区分，应该用分面解决。

几何对象

直方图就相当于分组计数再加上条形的几何对象。

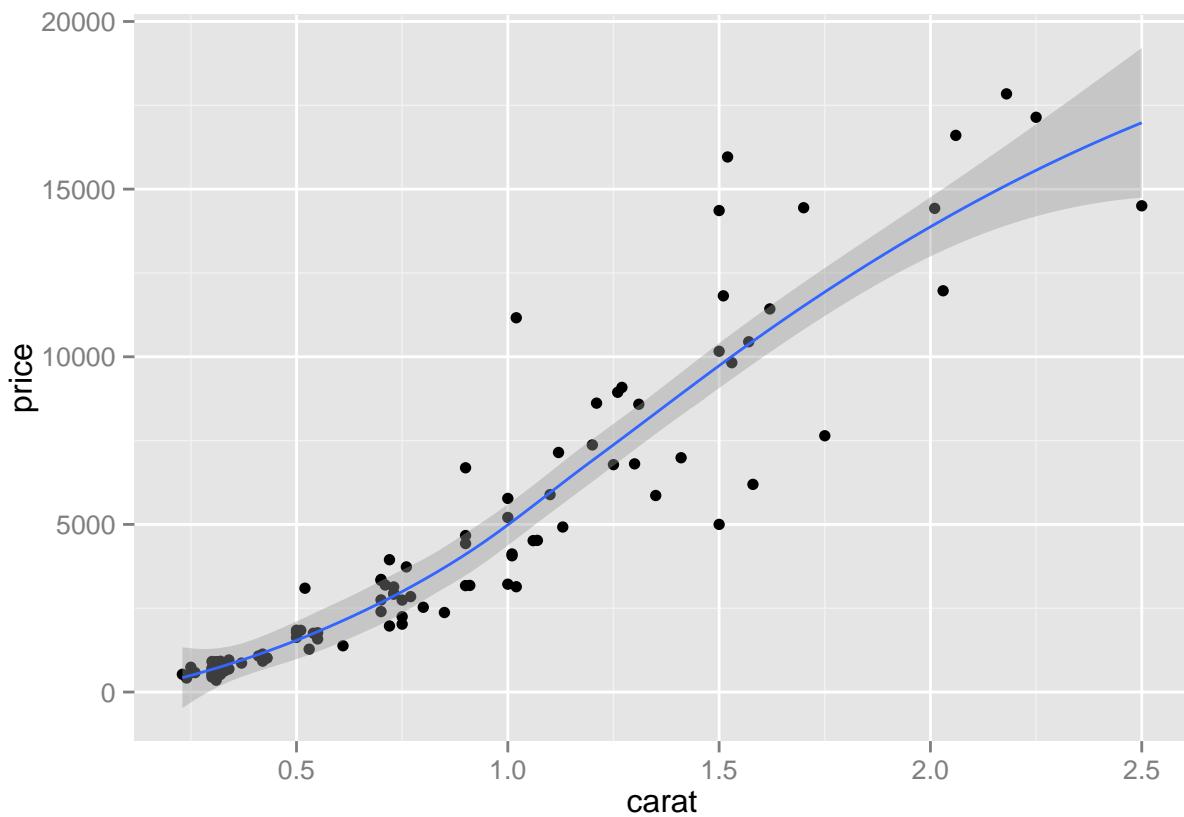
- geom=“point” 可以绘制散点图
- geom=“smooth” 将拟合一条光滑曲线。
- geom=“path” 和 “line” 可以在数据点之间绘制连线。
- 对于连续变量，geom=“histogram” 绘制直方图，geom=“freqploy” 绘制频率多边形，geom=“density” 绘制密度曲线。如果不给 qplot() 传参数，默认就是直方图几何对象。

如果在散点图中有非常多的数据点，那么数据展示的趋势可能不明显，此时可以添加一条平滑曲线。可用 c() 函数将多个几何对象组成一个向量传递给 geom，对象将按顺序堆叠。

```
## 重量与价格的散点图中加入了平滑曲线。左图为 dsmall 数据集，右图为完整数据集。
```

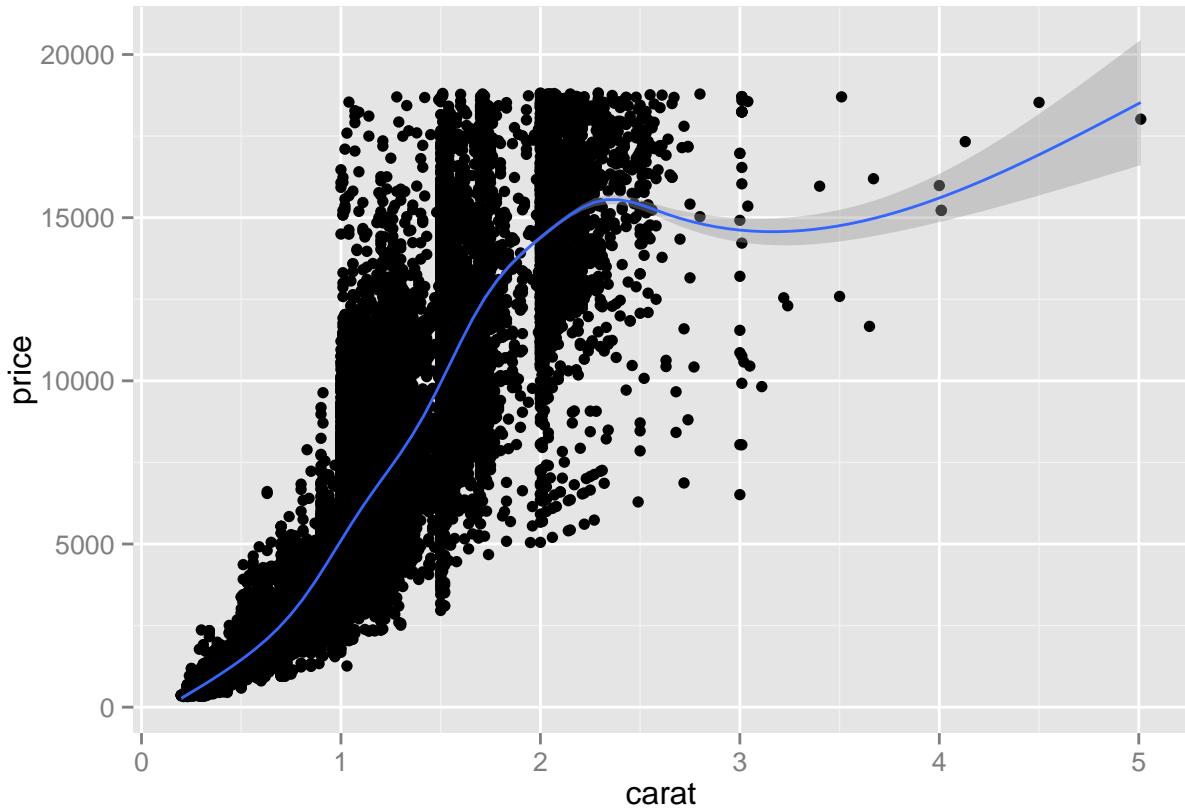
```
par(mfrow=c(1,2))
qplot(carat, price, data = dsmall, geom = c("point", "smooth"))
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method'
```



```
qplot(carat, price, data = diamonds, geom = c("point", "smooth"))
```

```
## geom_smooth: method="auto" and size of largest group is >=1000, so using gam with form
```

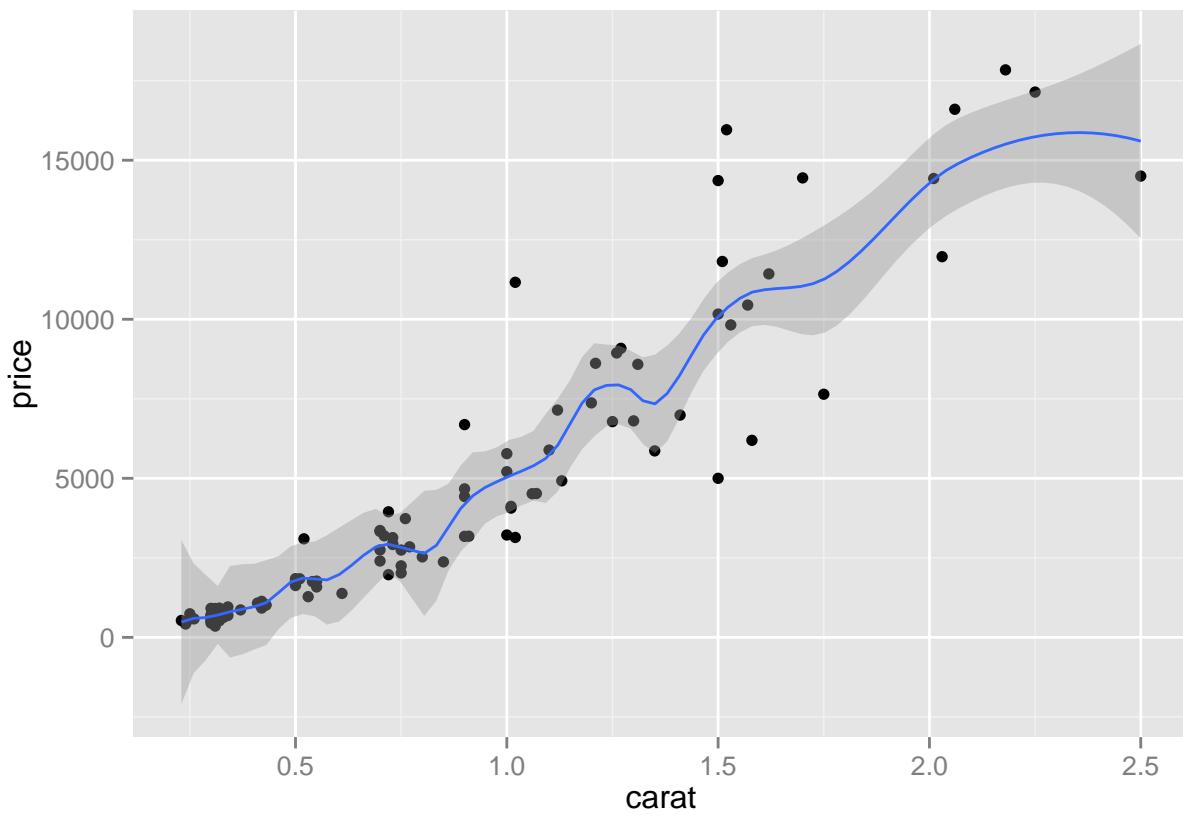


利用 method="loess" 参数可以设置许多不同的平滑器，当 n 较小时，使用的是局部回归的方法，可以?loess 查询帮助信息。曲线的平滑程度有 span 参数设定，越大越平滑。

```
## span 参数的作用。左图是 span=0.2，右图是 span=1。
```

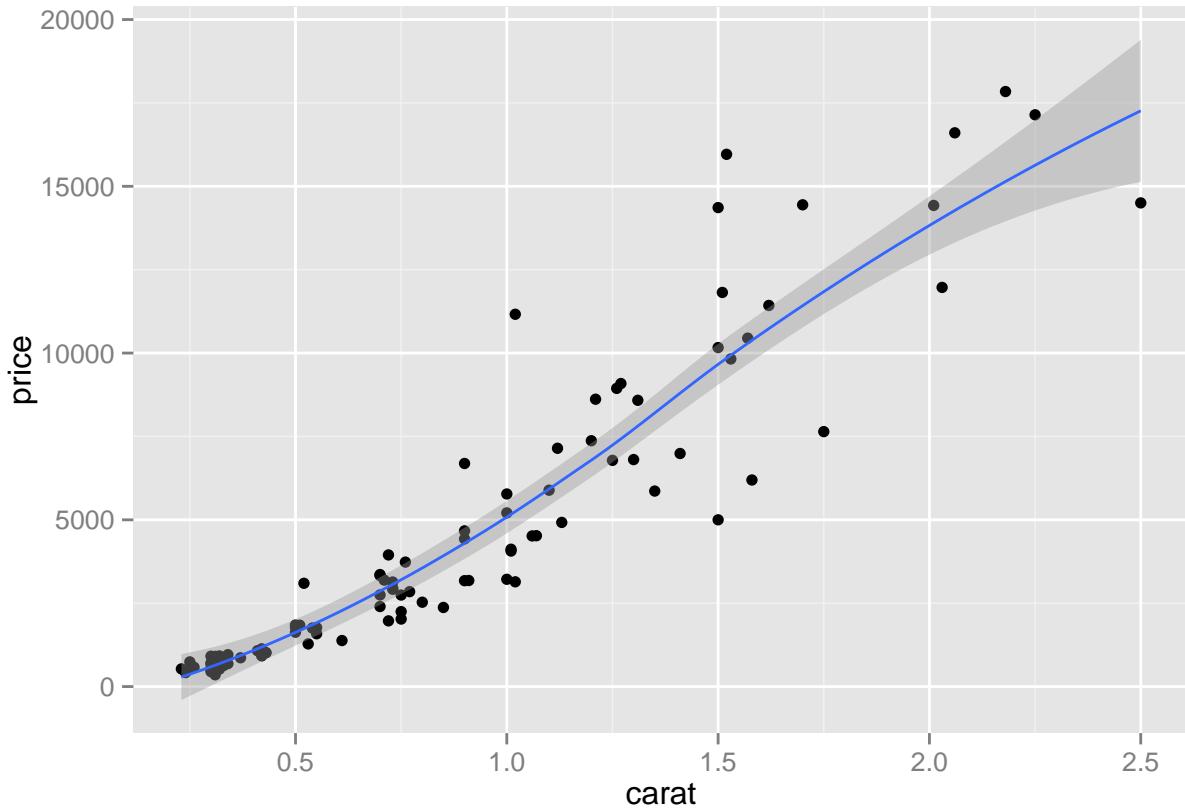
```
par(mfrow=c(1, 2))
qplot(carat, price, data = dsmall, geom = c("point", "smooth"), span = 0.2)
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'me
```



```
qplot(carat, price, data = dsmall, geom = c("point", "smooth"), span = 1)
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'me
```

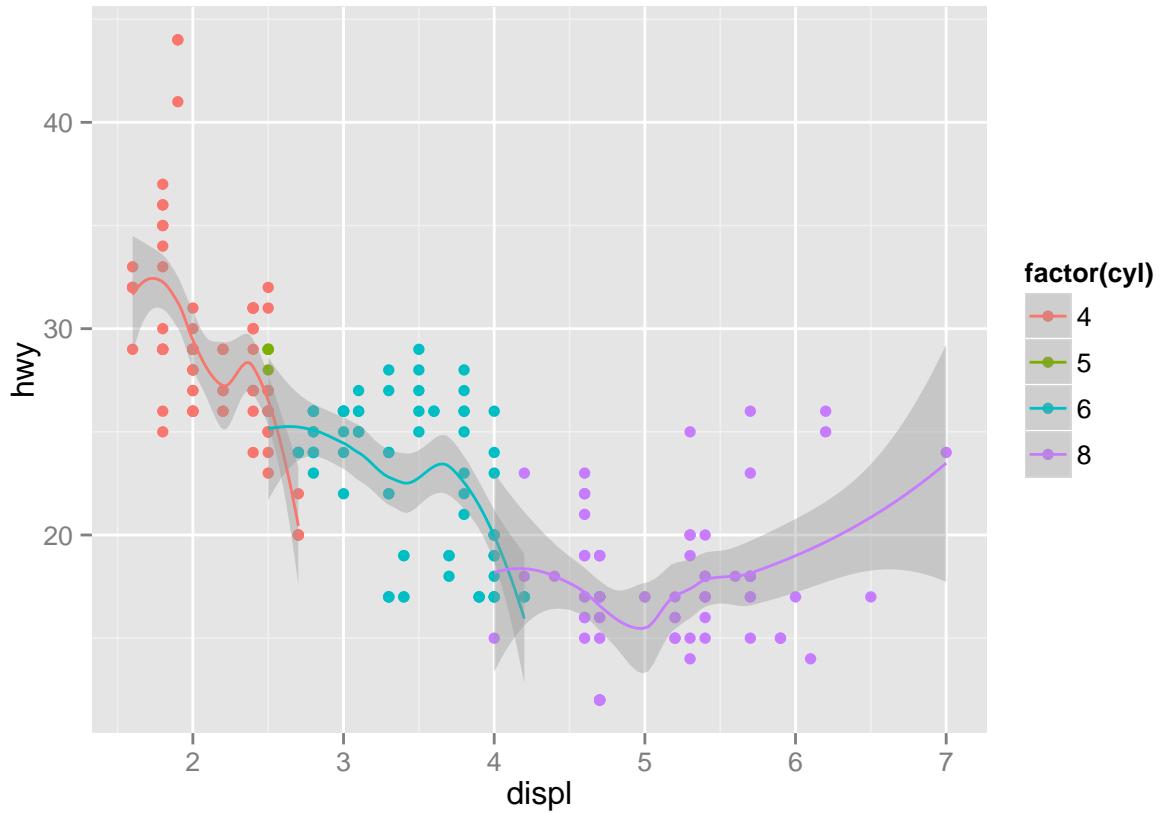


图层实例

下面用 ggplot2 包内带的汽车测试数据 (mpg) 来举个例子，用到的三个变量分别是发动机容量 (displ)、高速公路上的每加仑行驶里数 (hwy)、汽缸数目 (cyl)。首先加载 ggplot2 包，然后用 ggplot 定义第一层即数据来源。其中 aes 参数非常关键，它将 displ 映射到 X 轴，将 hwy 映射到 Y 轴，将 cyl 变为分类数据后映射为不同的颜色。然后使用 + 号添加了两个新的图层，第二层是加上了散点，第三层是加上了 loess 平滑曲线。

```
library(ggplot2)
p <- ggplot(data=mpg, aes(x=displ, y=hwy, colour=factor(cyl)))
p + geom_point() + geom_smooth()
```

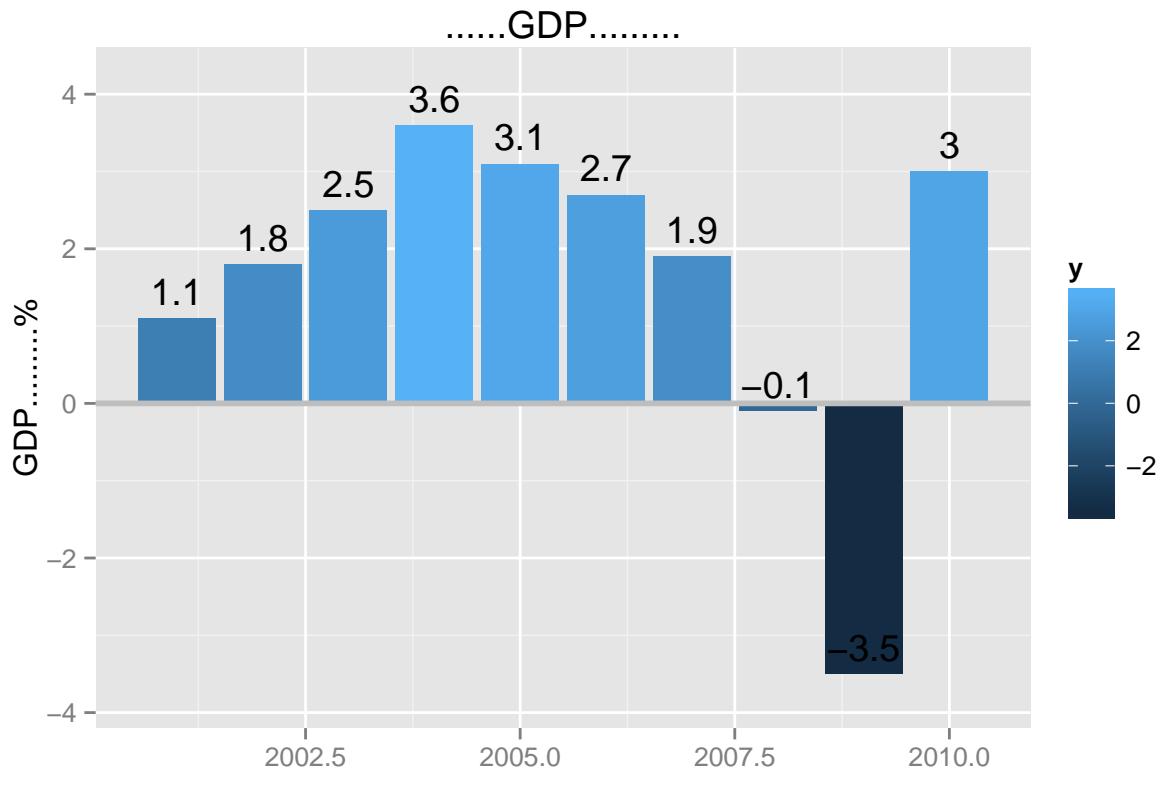
```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'me
```



条状图

`geom_bar` 是绘制条状几何对象，所以也可以用不经汇集的原始数据进行绘图。下面我们用 2001 到 2010 年间的美国 GDP 增长率举个例子。

```
y=c(1.1,1.8,2.5,3.6,3.1,2.7,1.9,-0.1,-3.5,3.0)
x=2001:2010
data=data.frame(x,y)
p=ggplot(data,aes(x,y,fill=y))
p+geom_bar(stat="identity")+
  geom_abline(intercept = 0, slope = 0,size=1,colour='gray')+
  geom_text(aes(label=y),hjust=0.5, vjust=-0.5 )+
  scale_y_continuous(limits=c(-3.8,4.2))+
  labs(x='年份', y='GDP 增长率%')+
  ggtitle(" 美国 GDP 增长率")
```



时间序列

ggplot2 包也能对时间序列数据绘图，但在处理上需要有些注意的地方。下面我们以上证指数为例进行作图，首先利用 quantmod 包从 yahoo 数据源获取从 1997 年以来的数据，存于变量 SSEC 中，抽取收盘数字，然后分别提取时间数据和指数数值，绘图结果如下图。

```
library(quantmod)
```

```
## Loading required package: xts
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
## 
##     as.Date, as.Date.numeric
##
## Loading required package: TTR
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
library(ggplot2)
getSymbols('^SSEC', src='yahoo', from = '1997-01-01')
```

```
## As of 0.4-0, 'getSymbols' uses env=parent.frame() and
```

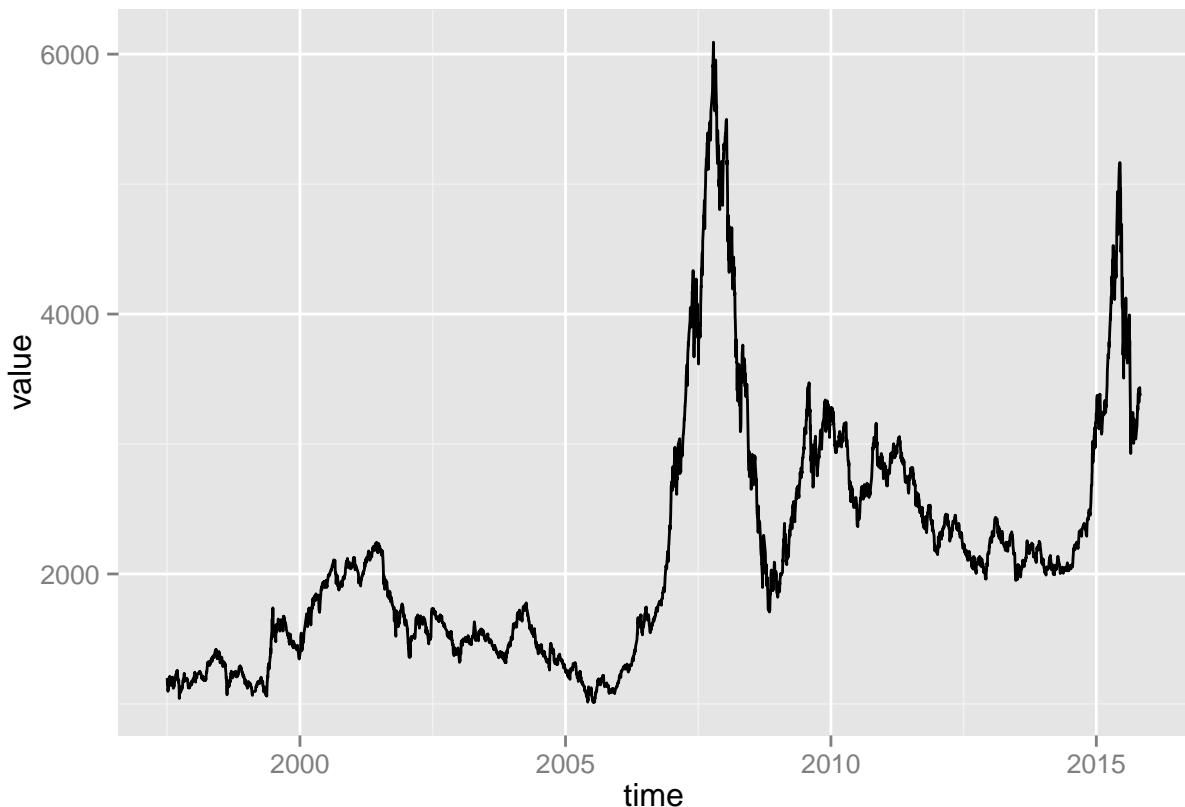
```

## auto.assign=TRUE by default.
##
## This behavior will be phased out in 0.5-0 when the call will
## default to use auto.assign=FALSE. getOption("getSymbols.env") and
## getOptions("getSymbols.auto.assign") are now checked for alternate defaults
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for more details.

## [1] "SSEC"

close <- (C1(SSEC))
time <- index(close)
value <- as.vector(close)
p <- ggplot(data.frame(time,value),aes(time,value))
p + geom_line()

```



ggplot 语法突破

简介

如果不深入研究底层语法，你可以选择只是用 qplot(), 但是这样你将永远无法掌握 ggplot 的精髓。

无论是对于普通用户还是对于统计图形开发人员，图层语法都是非常有用。对于普通用户，它使得图形的重复更新变得简单，每次只需更新一个特征；对于开发人员而言，图层语法使得向 ggplot2 里添加新功能变得更加方便。

耗油量数据

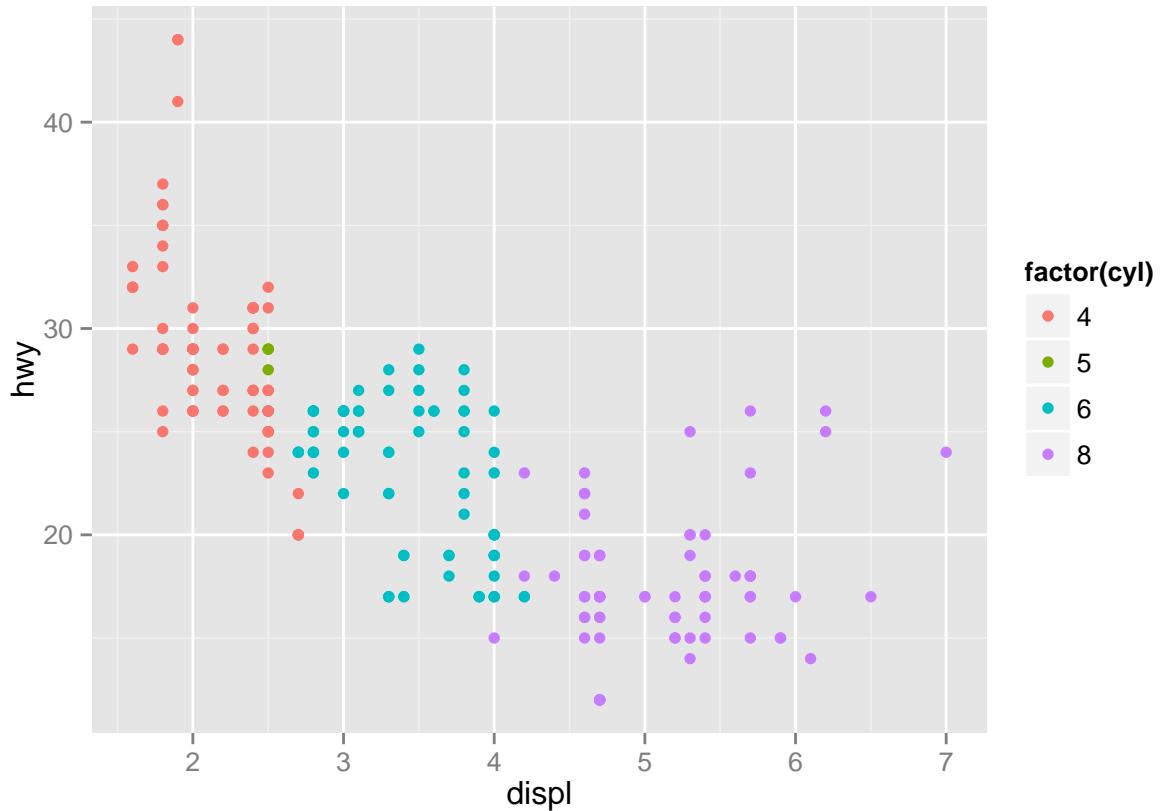
耗油量数据集 mpg，记录了美国 1999 年和 2008 年的部分汽车的制造厂商、型号、类别、引擎大小、传动系和耗油量等信息。包含 38 种型号的汽车。数据源自 EPA 燃油经济性网站。cty 和 hwy 分别记录了城市和高速公路驾驶时的耗油量(英里每加仑 mpg)，displ 表示发动机排量。

需要研究的问题：引擎大小和耗油量有什么关系？是不是某些制造商比其他制造商更关注汽车耗油量？耗油量在过去十年中有没有明显的增加？

绘制散点图

绘制发动机排量和高速公路每加仑行驶的英里数的散点图，点的颜色是由第三个变量汽缸数 cylinders 的数目决定的。先用 qplot 来作这幅图。

```
library(ggplot2)
## 发动机排量 (以升为单位 displ) 对高速公路耗油量 (英里每加仑
## hwy) 散点图。点
## 根据汽缸数目着色。该图可以发现影响燃油经济性最重要的因素：发动机排量大
## 小。
qplot(displ, hwy, data = mpg, colour = factor(cyl))
```



图形属性与数据的映射

散点图中每一个观测数据都用一个点表示，点的位置由两个变量的值决定。每个点不仅有横坐标和纵坐标还有大小颜色和形状等属性，称之为图形属性 aesthetics，每个图形属性都可以映射为一个变量或者设定成一个常数。上例中

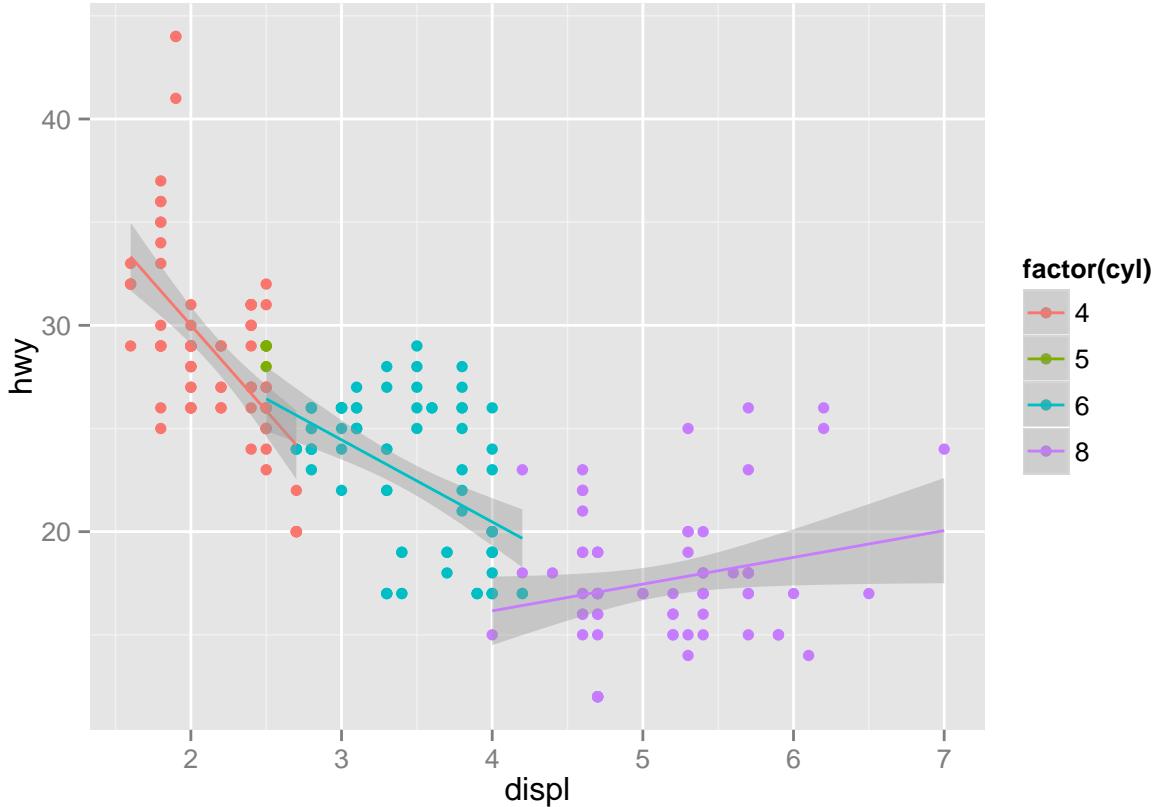
点的大小和形状没有指定映射的变量，使用就是默认值。

除了用点来表示数据，我们也可以用线和条几何对象来表示数据。几何对象决定了图形的类型，只含有一种几何对象的图通常都有特定的名字。由多种几何对象组合而成的更复杂的图形通常没有特定的名字，我们需要对其重新描述。在原有散点图基础上添加了分组回归线。

```
## 更复杂的图形一般没有特定的名称。这幅图在上图的基础上对每个
```

```
## 组添加了回归线。这个图应该叫什么名字呢？
```

```
qplot(displ, hwy, data = mpg, colour = factor(cyl)) + geom_smooth(data = subset(mpg, cyl !
```



标度变换

一些数据对电脑而言没有意义，需要把它们从数据单位(升，加仑，汽缸数)转换成电脑可以识别的物理单位(像素或颜色)，这个转换过程称之为标度变换 scaling，虽然转换后的数据对我们而言可能没有意义，但对于电脑而言却是可识别的，颜色用 6 个字母组成的十六进制字符表示，大小和形状分别用数字和整数来表示。位置变换非常简单，默认线性变换，只需将数据的范围线性映射到 $[0,1]$ 区间即可，因为 ggplot2 的绘图系统调用 grid 包会处理好最终的转换细节，所以用 $[0,1]$ 而不是精确的像素值。最后一步是如何根据点的位置 (x,y) 来确定它在图中的位置，这是由坐标系决定的，称为 coord，大多数情况下使用笛卡尔坐标系。颜色变换过程稍微复杂点，因为将得到一个非数字的结果。颜色可以看成由三个组件组成，三种颜色空间，颜色的标度转换相当于将数据的值映射到这个空间，映射方法有很多。最后还需要对图形进行渲染，生成能在屏幕上展示的图形对象。

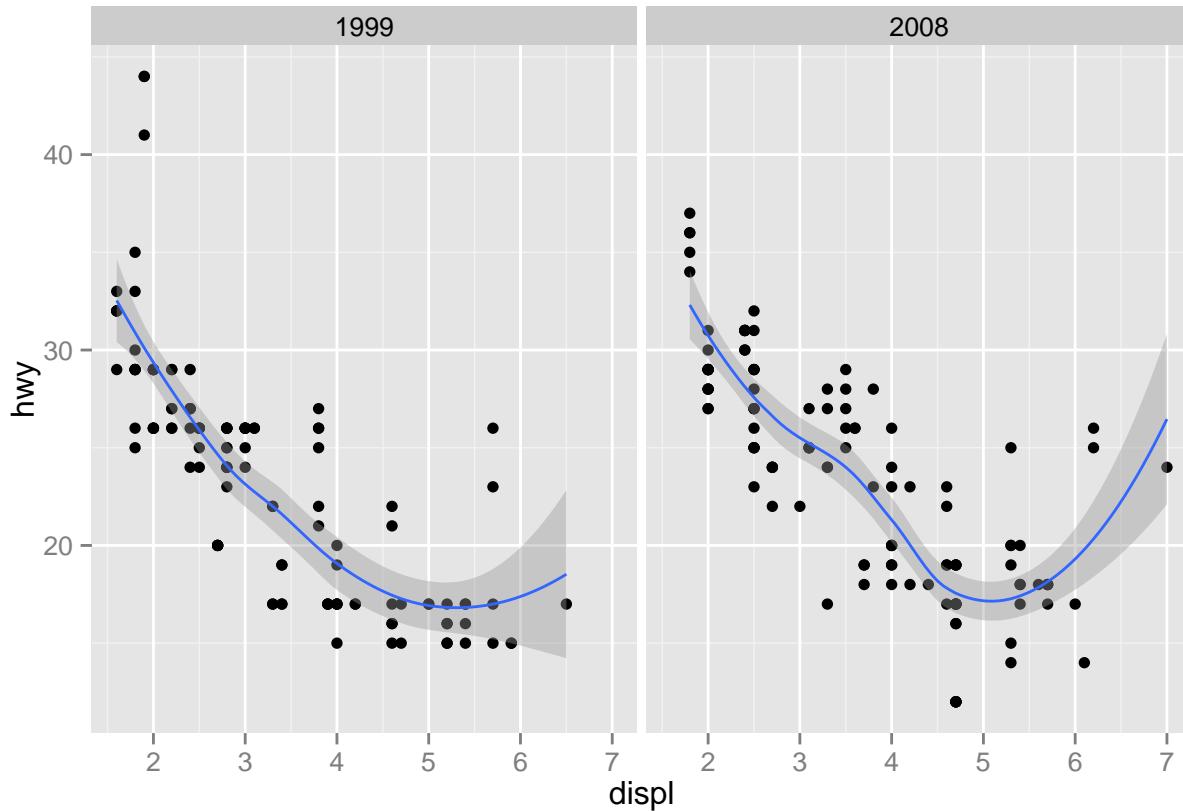
更复杂的图形实例

这幅图添加了三个新的组件，分面，多个图层和统计量。分面和图层将数据切割成多个小的数据集，即每个图层的每个分面面板都含有一个小的数据集。可以想象成三维矩阵：分面面板形成一个 2 维网格，图层在第 3 维的方向上

叠加。本例中所有图层的数据都是一样的，但是就一般而言，我们可以在不同的图层里使用不同的数据集。

```
## 一个含有分面和多个图层的复杂图形  
qplot(displ, hwy, data = mpg, facets = . ~ year) + geom_smooth()
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method' argument to force SVD or LOESS  
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method' argument to force SVD or LOESS
```



平滑曲线层与散点层的不同点在于它没有展示原数据，而是展示统计变换后的数据。平滑曲线拟合了一个穿过数据中间位置的平滑曲线。

图层

图层的作用是生成在图像上可以被人感知的对象，一个图层由四个部分组成：

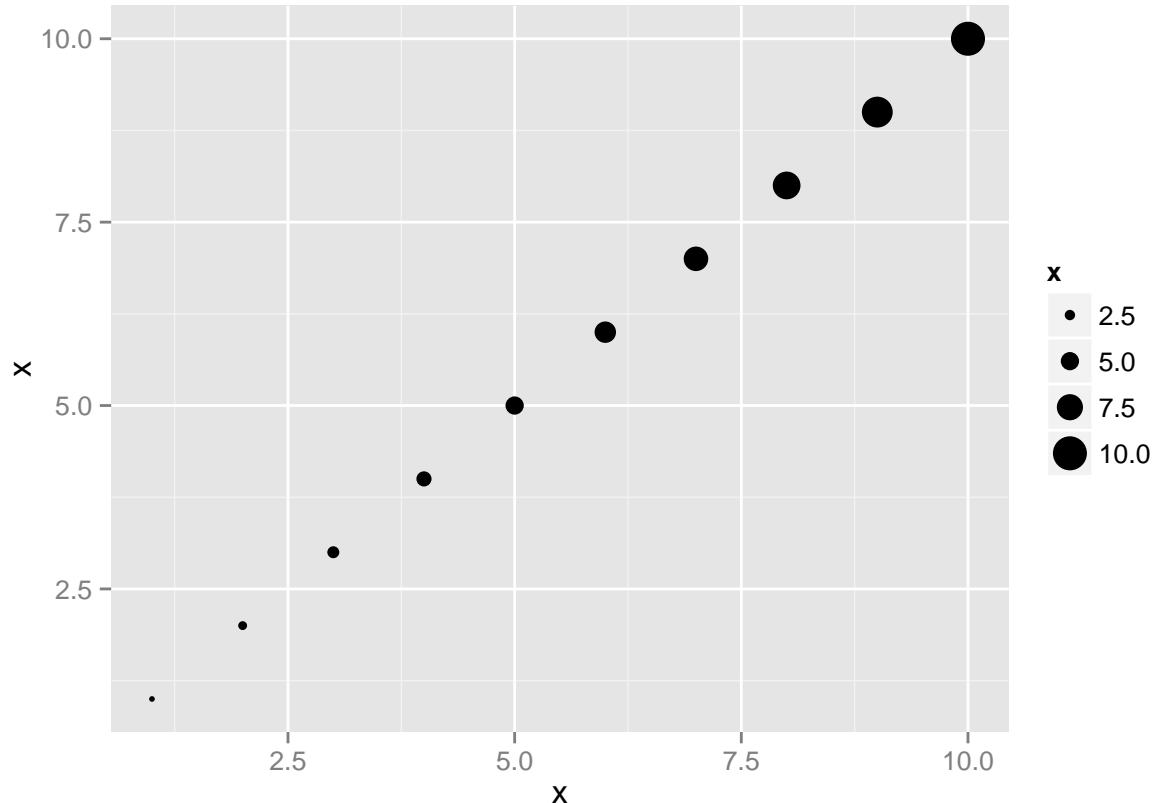
- 数据和图形属性映射；
- 一种统计变换：比如求均值，求方差等，当我们需要展示出某个变量的某种统计特征的时候，需要用到统计变换
- 一种几何对象； * 一种位置调整方式。

标度

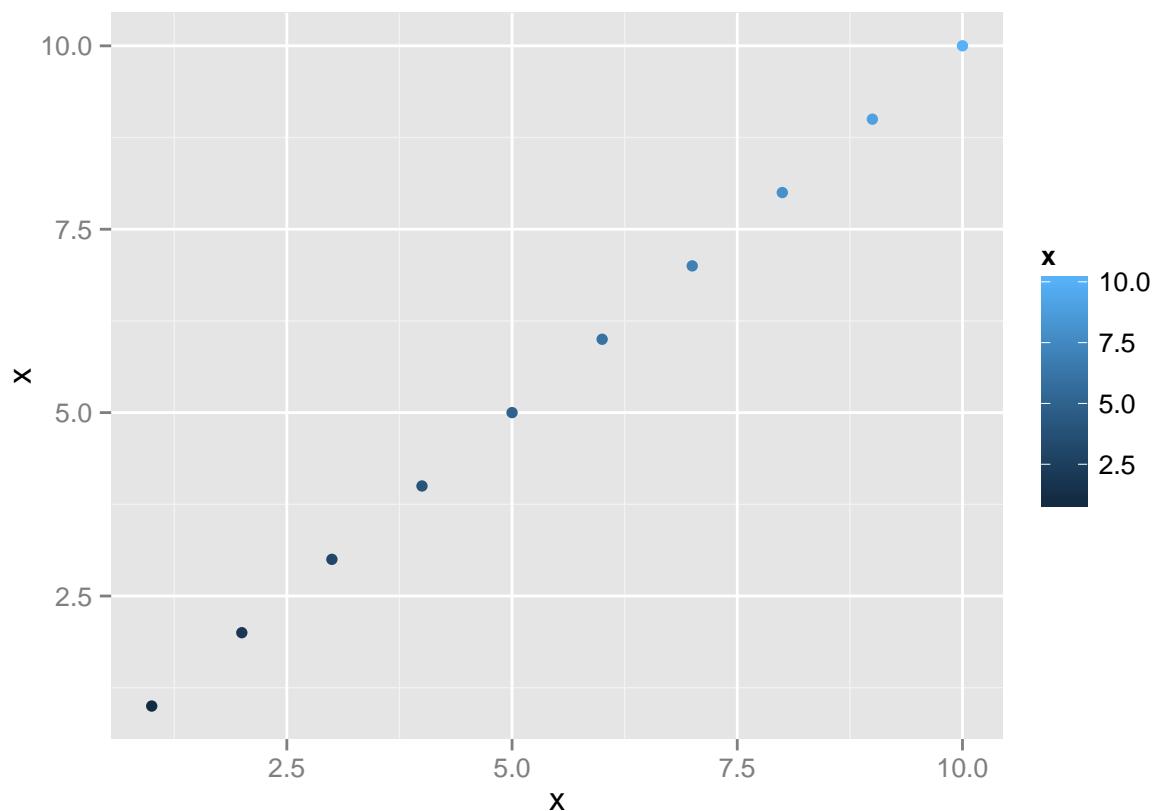
标度控制着数据到图形属性的映射，并且图形上所用的每一个图形属性都对应着一个标度。每个标度都作用于图形中的所有受，以确保从数据到图形属性映射的一致性。一个标度就是一个含一组参数的函数，它的逆也是如此。其

逆函数被用来绘制参照对象，通过参照对象你才能读出图里隐含的信息，参照对象可以是坐标轴或图例(其他标度)，大多数映射都有唯一的逆函数，但有些不是。

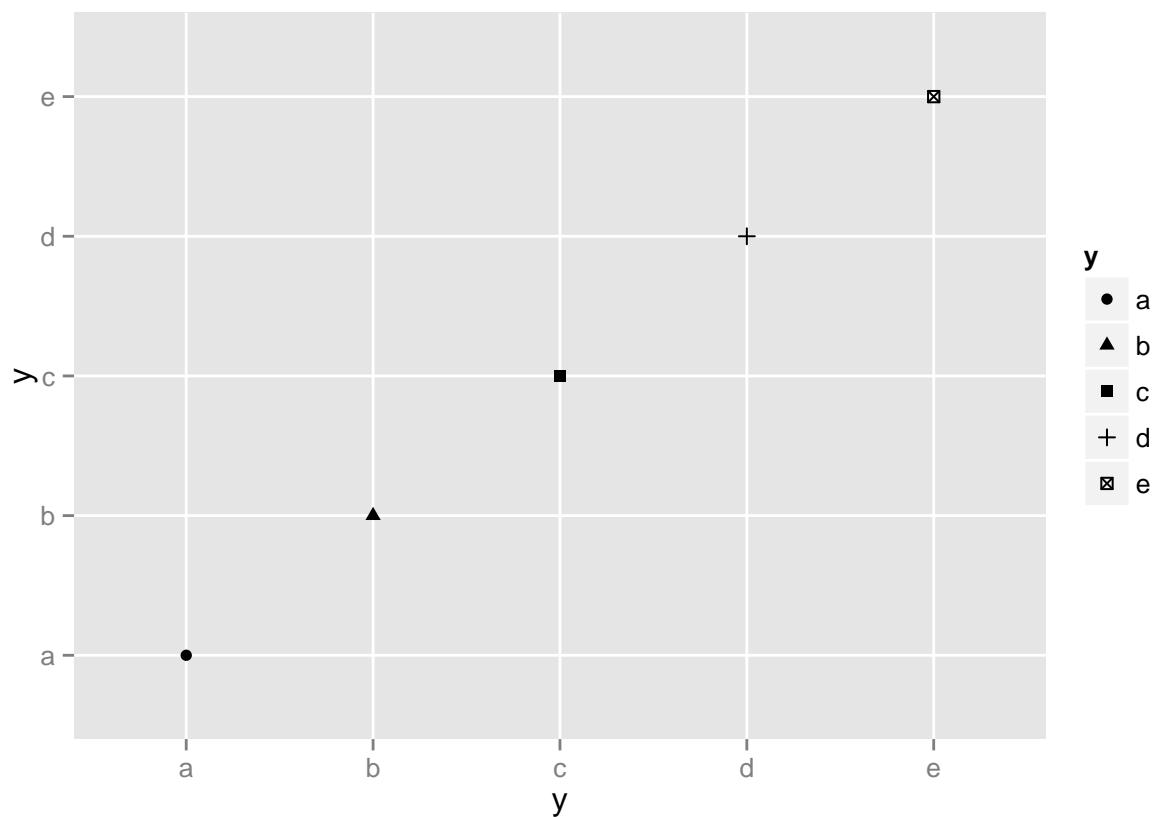
```
## 四种不同标度的图例。从左到右依次是：连续型变量映射到大小和颜色，离散型变  
## 量映射到形状和颜色。  
par(mfrow=c(2,2))  
x <- 1:10  
y <- factor(letters[1:5])  
qplot(x, x, size = x)
```



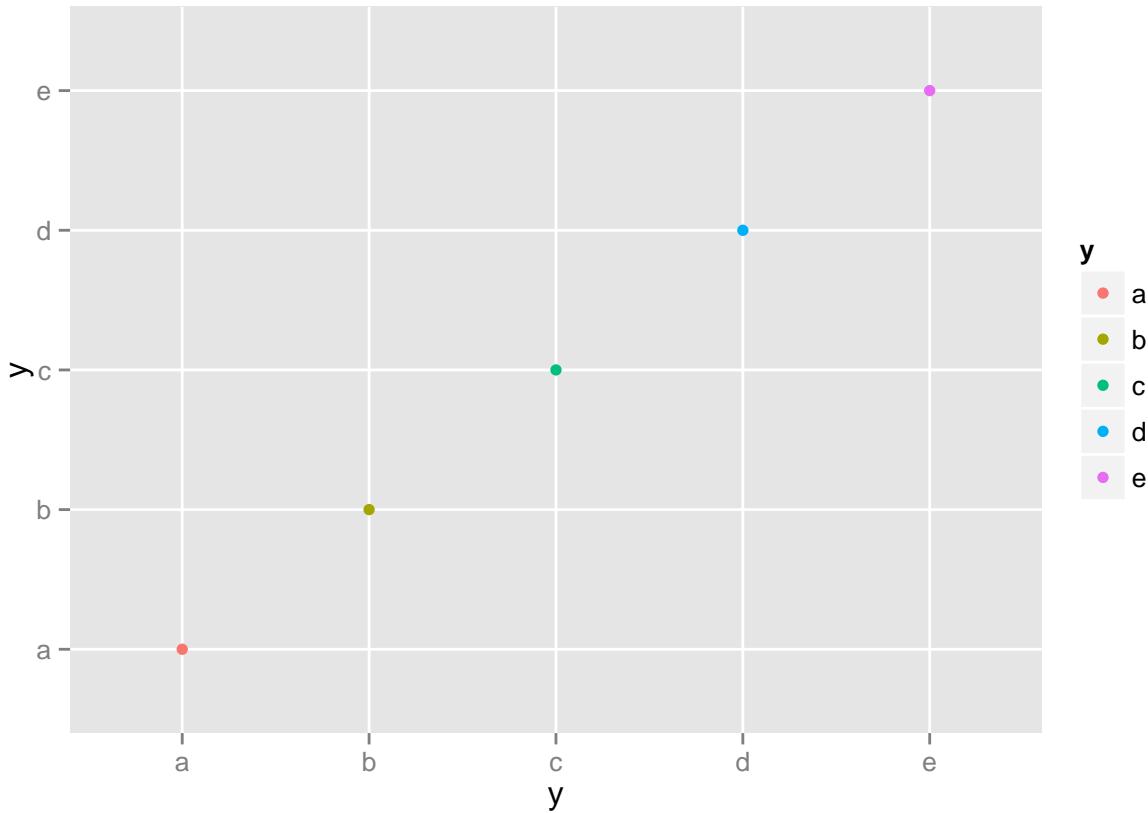
```
qplot(x, x, 1:10, colour = x)
```



```
qplot(y, y, 1:10, shape = y)
```



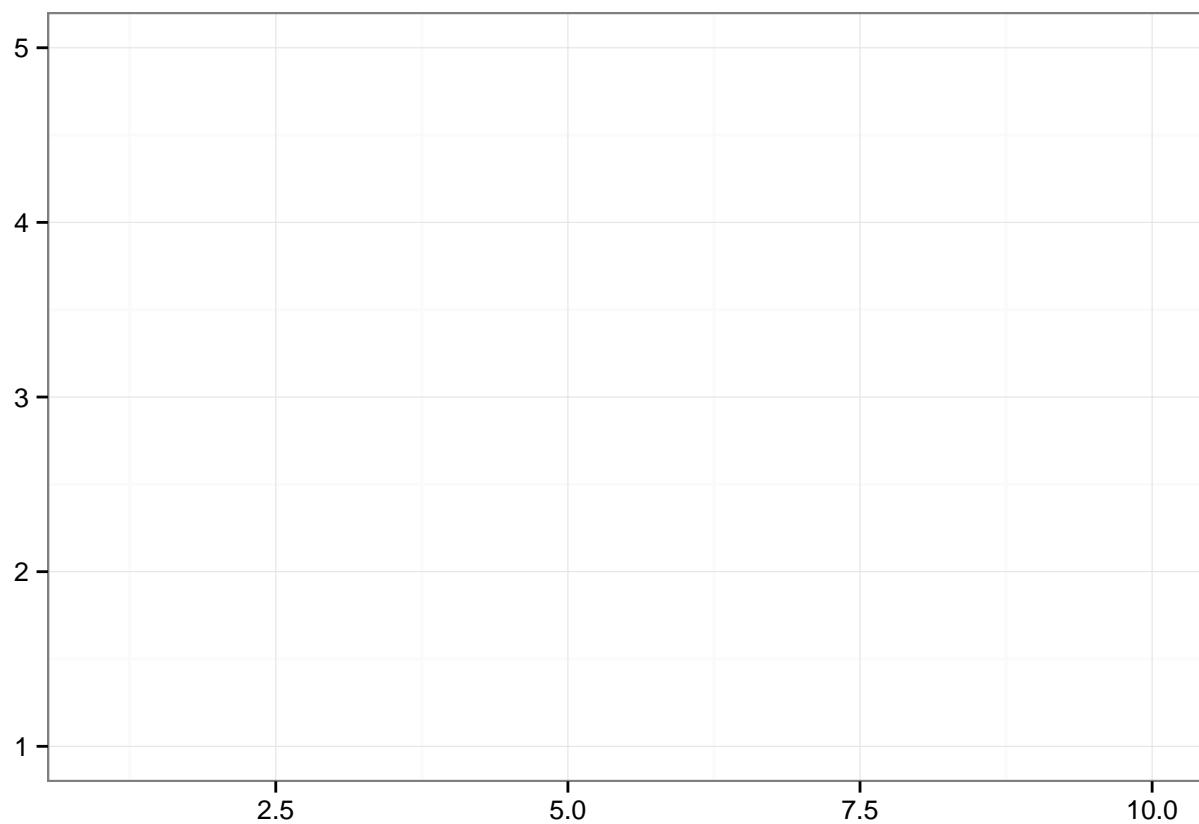
```
qplot(y, y, 1:10, colour = y)
```



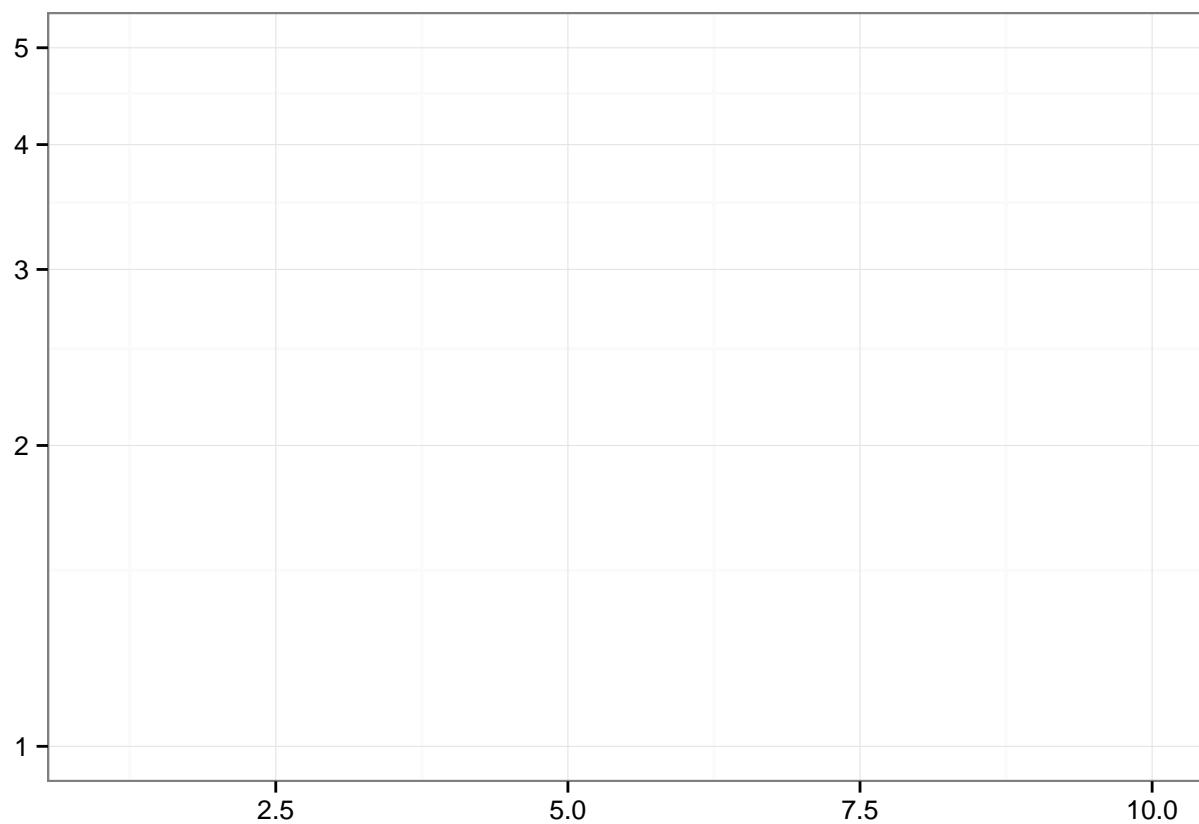
坐标系

可将对象的位置映射到图形平面上，可以同时影响所以的位置变量。与标度不同，坐标系还可以改变几何对象的外观，如，极坐标系中，条形看起来像扇形。另外，标度变换时在统计变换之前执行的，而坐标系变换是在此之后执行的。

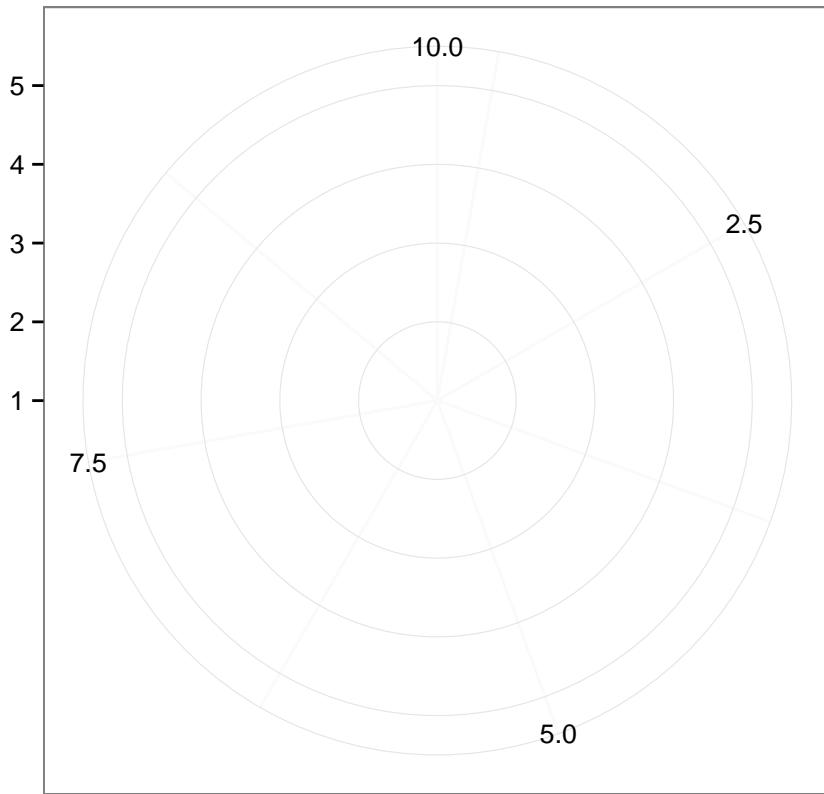
```
## 三种不同坐标系的坐标轴和网格线：笛卡尔 (Cartesian)、半对数 (semi-log) 和极
## 坐标系 (polar)。极坐标系展示了非笛卡尔坐标系的缺点：很难画好坐标轴。
x1 <- c(1, 10)
y1 <- c(1, 5)
p <- qplot(x1, y1, geom = "blank", xlab = NULL, ylab = NULL) + theme_bw()
p
```



```
p + coord_trans(y = "log10")
```



```
p + coord_polar()
```



数据结构

ggplot2 的图形语法通过一种非常简单直接的方法编码到 R 的数据结构中。一个图形对象就是一个包含数据，映射，图层，标度，坐标，和分面的列表。

```
p <- qplot(displ, hwy, data = mpg, colour = factor(cyl))
summary(p)

## # data: manufacturer, model, displ, year, cyl, trans, drv, cty, hwy,
##   fl, class [234x11]
## # mapping: colour = factor(cyl), x = displ, y = hwy
## # facetting: facet_null()
## -----
## # geom_point:
## # stat_identity:
## # position_identity: (width = NULL, height = NULL)

# 保存图形对象
save(p, file = "plot.rdata")
# 读入图形对象
load("plot.rdata")
# 将图片保存成 png 格式
ggsave("plot.png", width = 5, height = 5)
```

save() 把图形的缓存副本保存到磁盘，load() 函数重现该图，summary() 查看图的结构。

用图层构建图像

qplot() 局限性在于只能使用一个数据集合一组图形属性映射，解决这个问题的办法是使用图层。每个图层都有自己的数据集和图形属性映射，附加的数据元素可以通过图层添加到图形中。

创建绘图对象

如果想手动创建图形对象，就要用到 ggplot() 函数。该函数有两个主要的参数：数据和图形属性映射。只有在新添加的图层里设定了新参数时，默认值才会被修改。数据必须用数据框，参数映射的设定方法与 qplot 函数类似，只需将图形属性和变量名放到函数 aes() 的括号里即可。下面设定了一组默认映射，x 为 carat，y 为 price，color 为 cut，这个图形对象在加上图层之前是无法显示的，因此你现在什么都看不见。

```
library(ggplot2)
## 通过 ggplot 创建图形对象
p <- ggplot(diamonds, aes(carat, price, colour = cut))
```

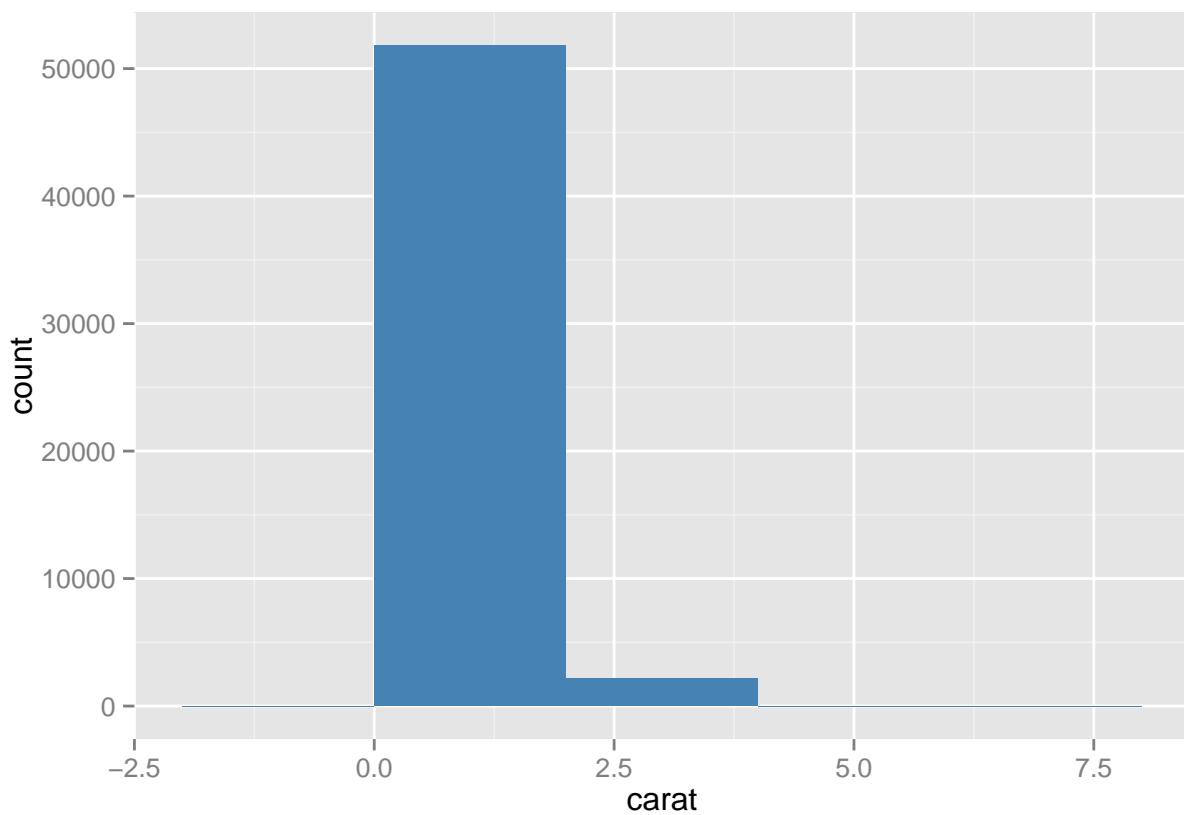
图层

最简易的图层只设定一个几何图形，这是数据可视化的一种方法。添加点几何对象形成了散点图。

```
## 添加“点”几何对象
p <- p + layer(geom = "point")
```

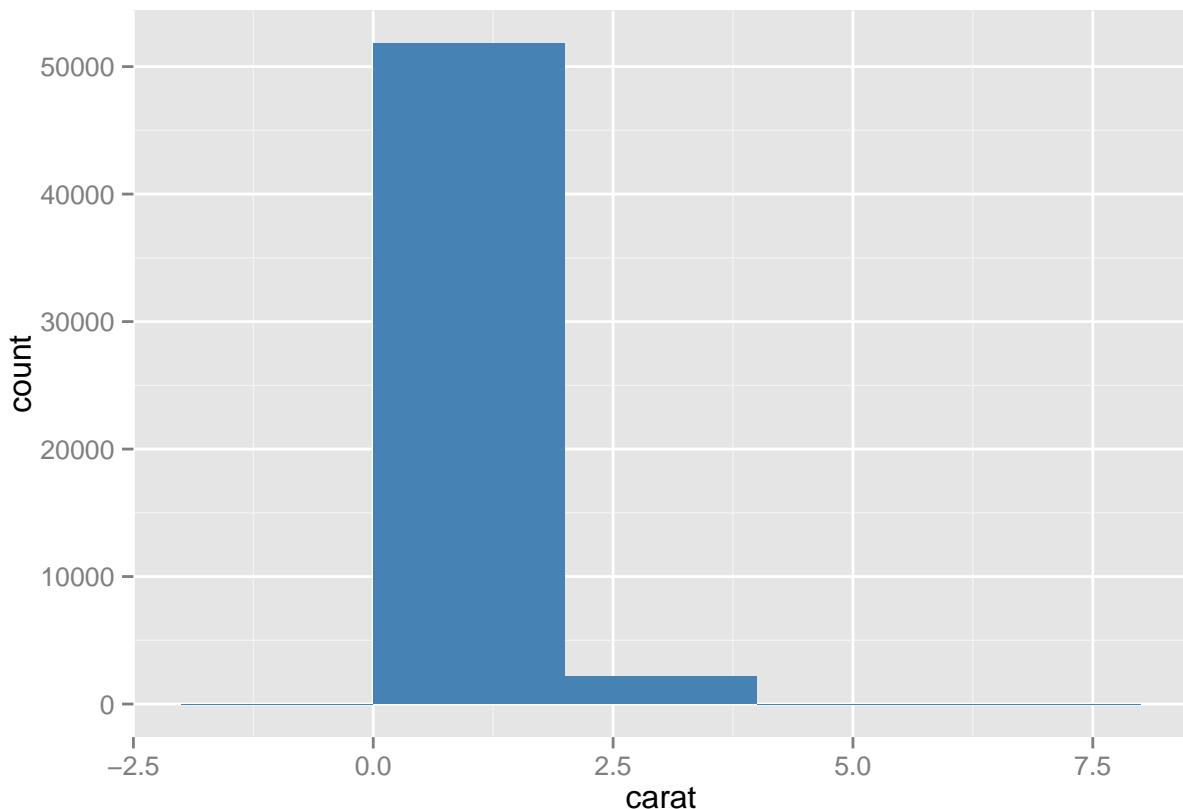
注意用“+”号添加图层。还有两个可选参数：统计变换和位置调整。

```
## 例：手动创建图形对象并添加图层
p <- ggplot(diamonds, aes(x = carat))
p <- p + layer(geom = "bar", geom_params = list(fill = "steelblue"), stat = "bin",
    stat_params = list(binwidth = 2))
p
```



上述图层参数设定得非常细致但是过于繁琐，可以用快捷函数来简化上面代码，因为每一个几何对象都对应着一个默认的统计变换和位置参数，而每一个统计变换都对应着一个默认的几何对象参数，所以对于一个图层只需设定 stat 或 geom 参数即可。下面代码生成同样的图层。

```
# 应用“快捷函数”，得到与上例相同的图形  
p + geom_histogram(binwidth = 2, fill = "steelblue")
```

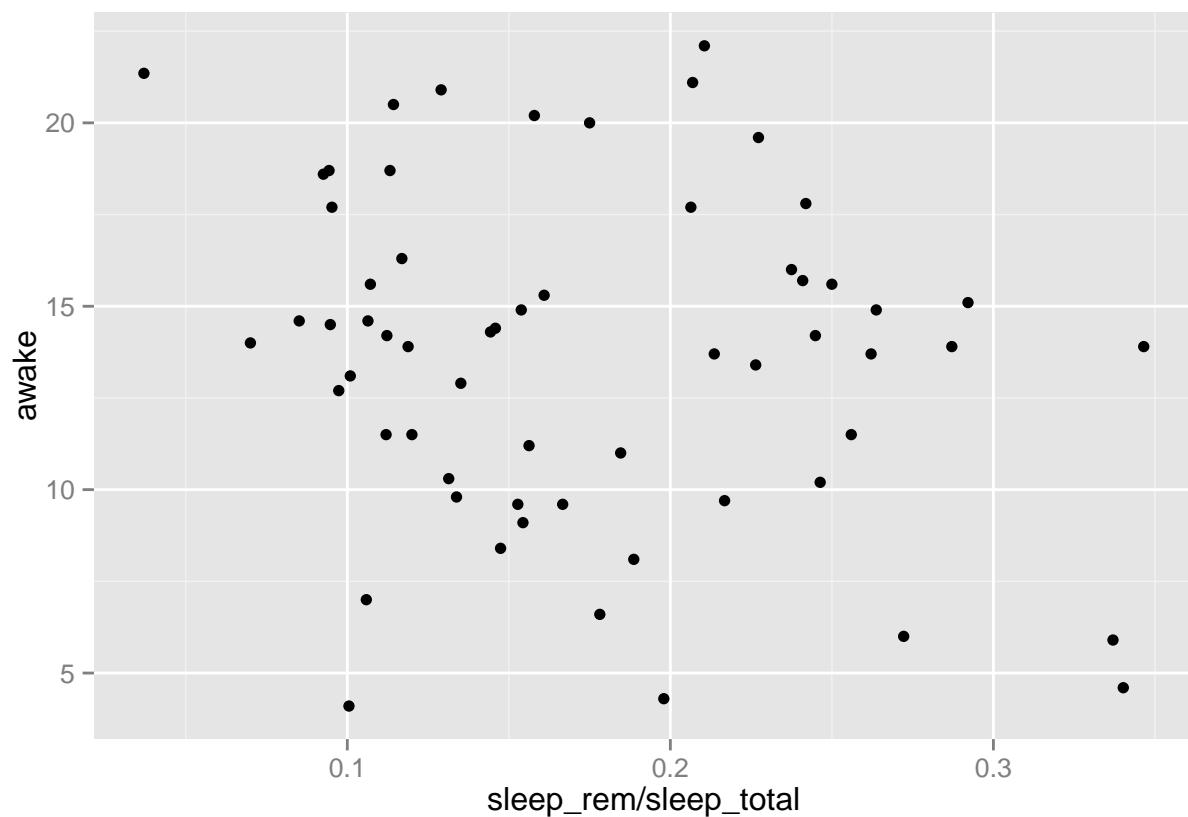


所有这类快捷函数都有相同的形式：`geom_` 或 `stat_` 开头。参数 `data` 和 `mapping` 在 `ggplot()` 函数和图层函数中的位置是相反的。因为我们在图形对象中一般先设定数据集，而在图层函数中大多是设定图形属性而不是数据集。建议写清参数名而不要依赖于参数的相对位置来设置参数，这样可以使得代码更具可读性，这也是本书遵循原则之一。

图层可以被添加到 `ggplot()` 和 `qplot()` 创建的图形对象上。实际上 `qplot()` 的绘图原理就是先创建图形对象然后再添加图层。

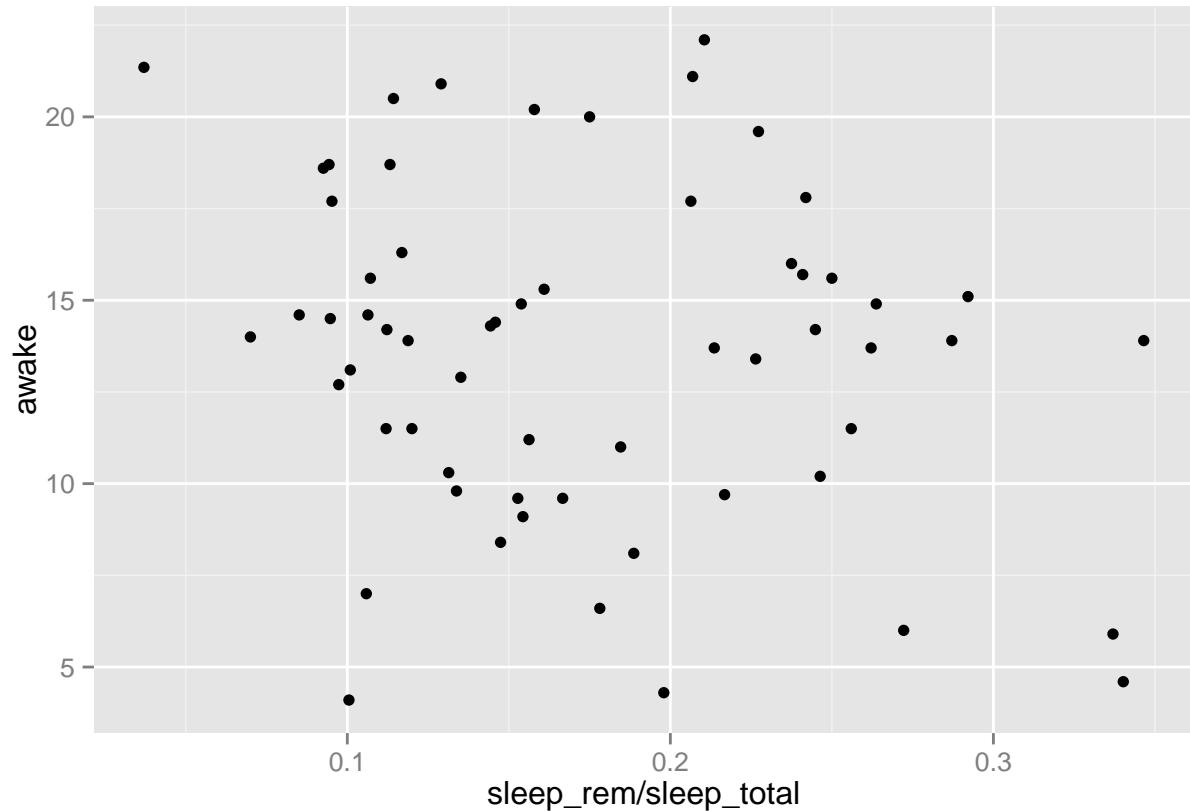
```
## 在用 ggplot 创建的图形对象上添加图层
ggplot(msleep, aes(sleep_rem/sleep_total, awake)) + geom_point()
```

```
## Warning: Removed 22 rows containing missing values (geom_point).
```



```
# 等价于  
qplot(sleep_rem/sleep_total, awake, data = msleep)
```

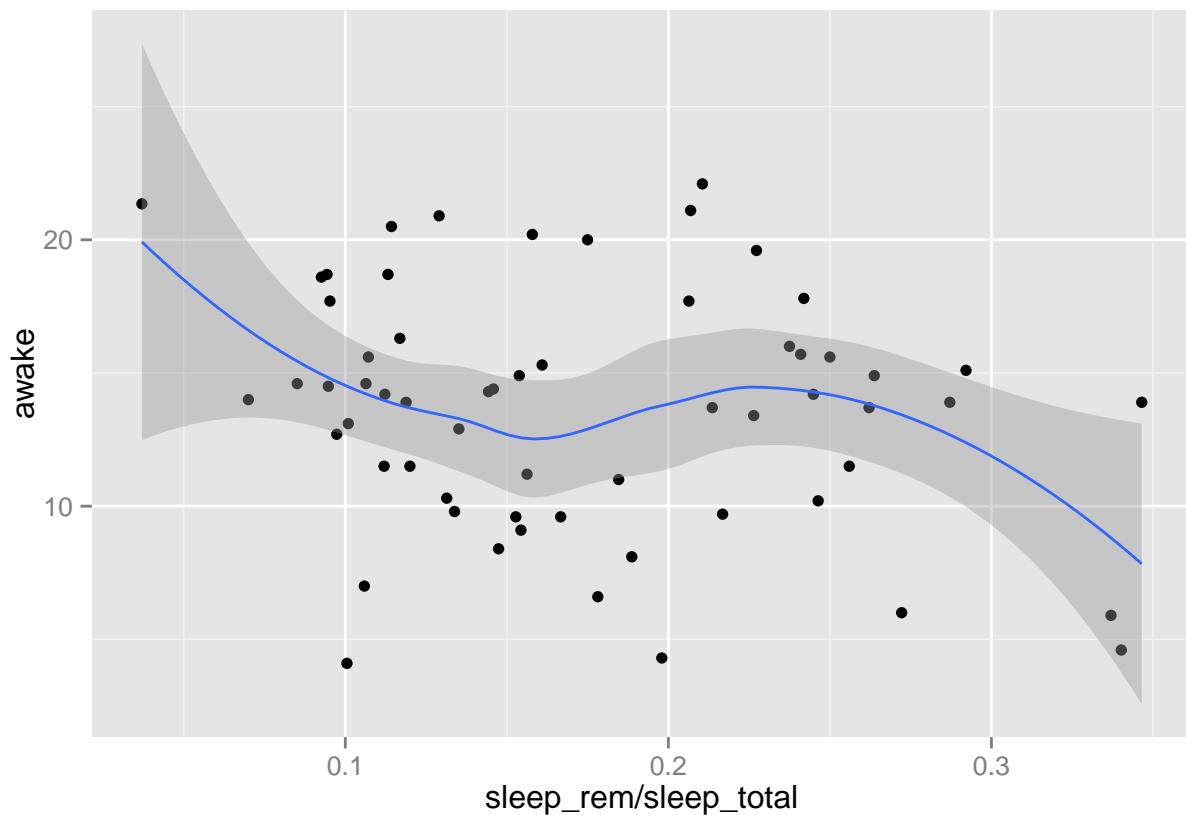
```
## Warning: Removed 22 rows containing missing values (geom_point).
```



```
# 也可以给 qplot 添加图层
qplot(sleep_rem/sleep_total, awake, data = msleep) + geom_smooth()

## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method = "loess"' if you want to force LOESS
## Warning: Removed 22 rows containing missing values (stat_smooth).

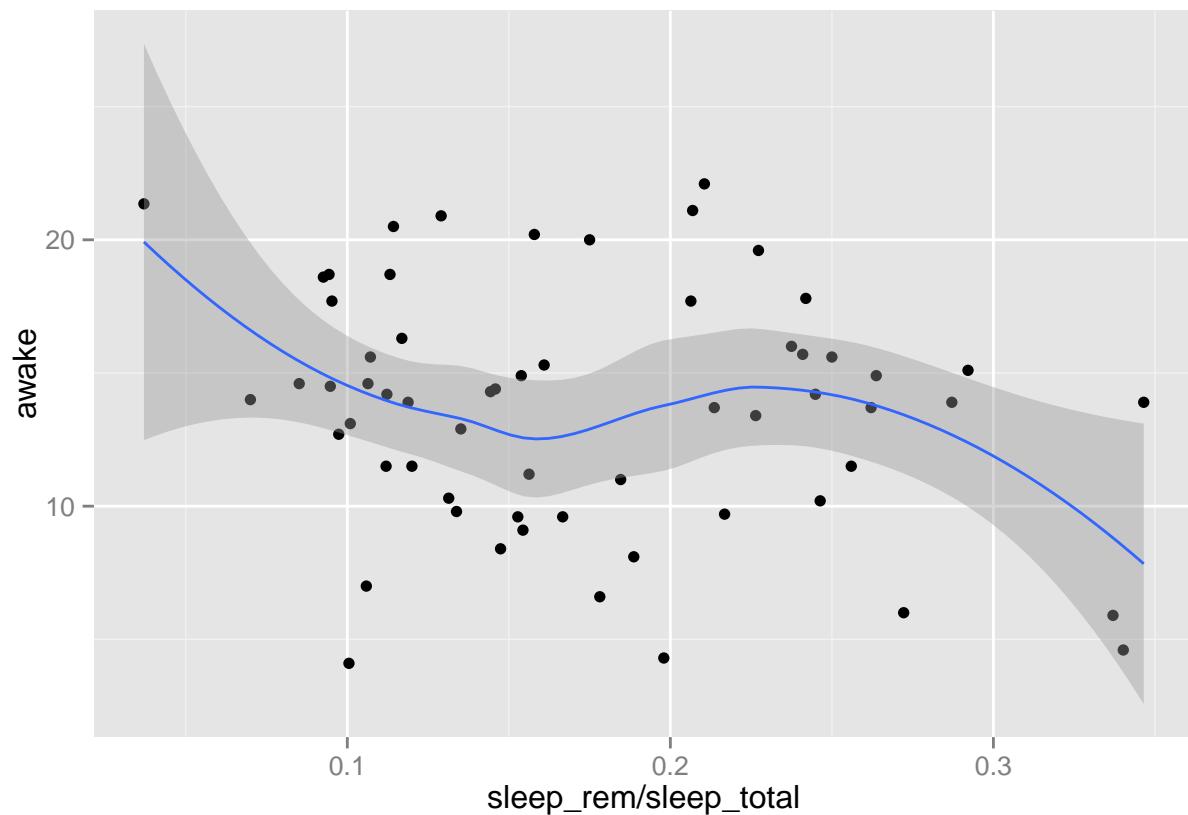
## Warning: Removed 22 rows containing missing values (geom_point).
```



```
# 等价于
qplot(sleep_rem/sleep_total, awake, data = msleep, geom = c("point", "smooth"))
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method = ...'
## Warning: Removed 22 rows containing missing values (stat_smooth).

## Warning: Removed 22 rows containing missing values (geom_point).
```

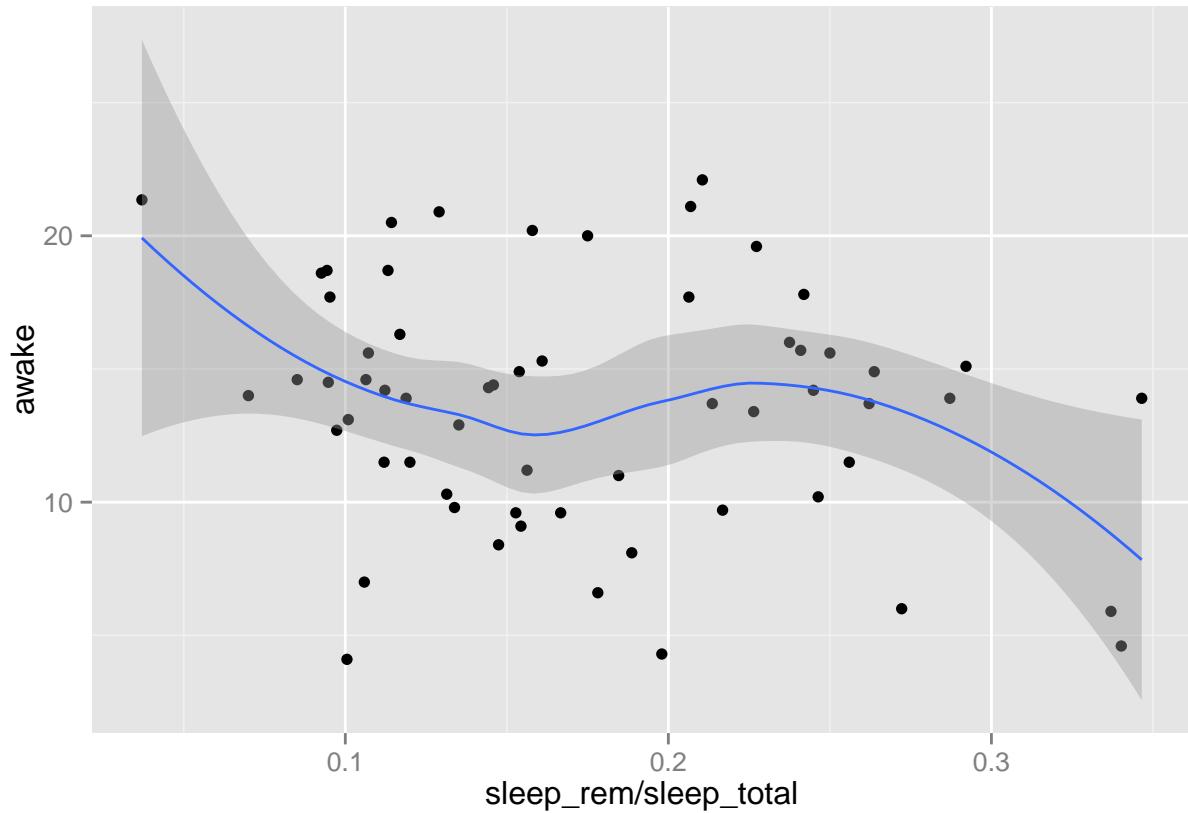


或

```
ggplot(msleep, aes(sleep_rem/sleep_total, awake)) + geom_point() + geom_smooth()

## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method = ...'
## Warning: Removed 22 rows containing missing values (stat_smooth).

## Warning: Removed 22 rows containing missing values (geom_point).
```



图形对象可以存储在一个变量里。summary 函数可以帮助我们查看图形对象的结构而不直接绘制图形。

```
## 例: summary 给出图形对象的默认设置和每个图层的信息
p <- ggplot(msleep, aes(sleep_rem/sleep_total, awake))
summary(p)
```

```
## data: name, genus, vore, order, conservation, sleep_total,
##   sleep_rem, sleep_cycle, awake, brainwt, bodywt [83x11]
## mapping: x = sleep_rem/sleep_total, y = awake
## faceting: facet_null()
```

```
p <- p + geom_point()
summary(p)
```

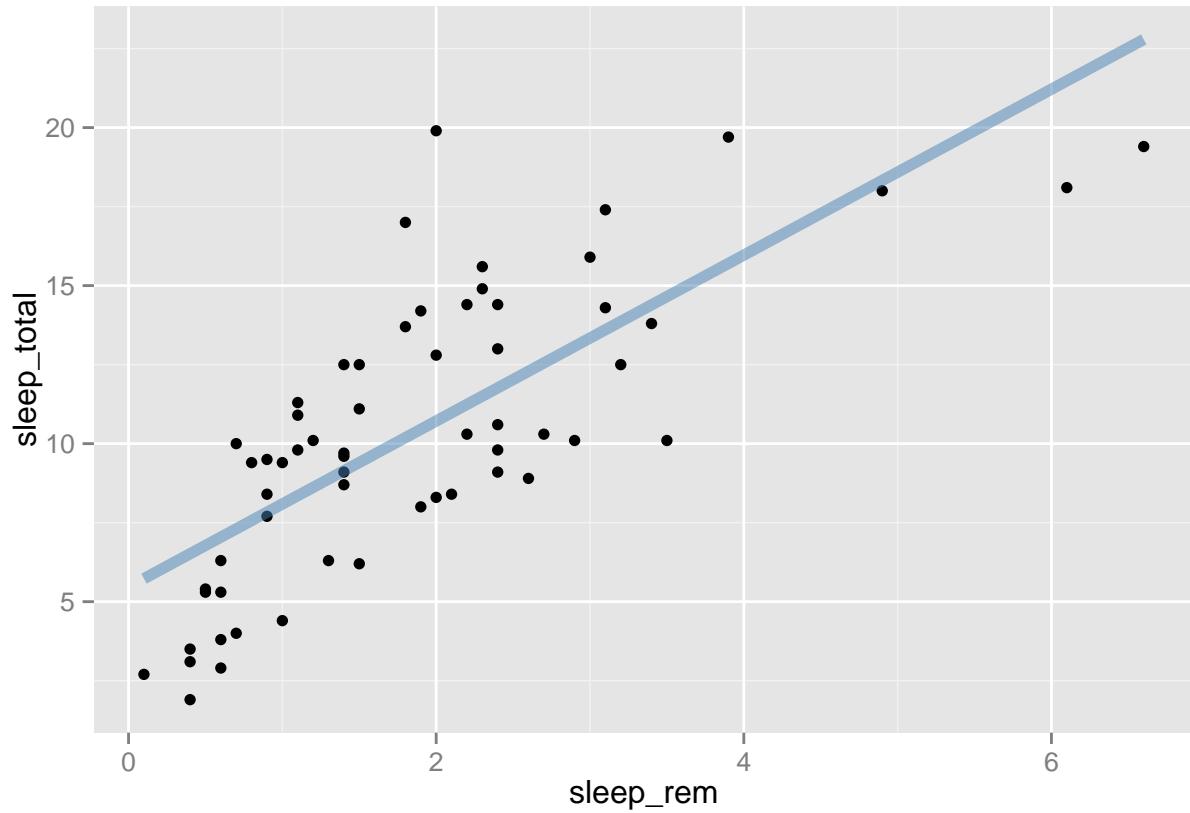
```
## data: name, genus, vore, order, conservation, sleep_total,
##   sleep_rem, sleep_cycle, awake, brainwt, bodywt [83x11]
## mapping: x = sleep_rem/sleep_total, y = awake
## faceting: facet_null()
## -----
## geom_point: na.rm = FALSE
## stat_identity:
## position_identity: (width = NULL, height = NULL)
```

图层是普通的 R 对象，所以可以存储到变量里，这有利于代码避繁就简。一组图形可以用不同的数据来进行初始化，然后加上相同的图层，如果后面想改变图层，只需修改一个地方即可。下面实例创建一个半透明深蓝色回归线的图层。

```
## 例：用不同的数据初始化后添加相同的图层
library(scales)
bestfit <- geom_smooth(method = "lm", se = F, colour = alpha("steelblue", 0.5),
size = 2)
qplot(sleep_rem, sleep_total, data = msleep) + bestfit
```

```
## Warning: Removed 22 rows containing missing values (stat_smooth).
```

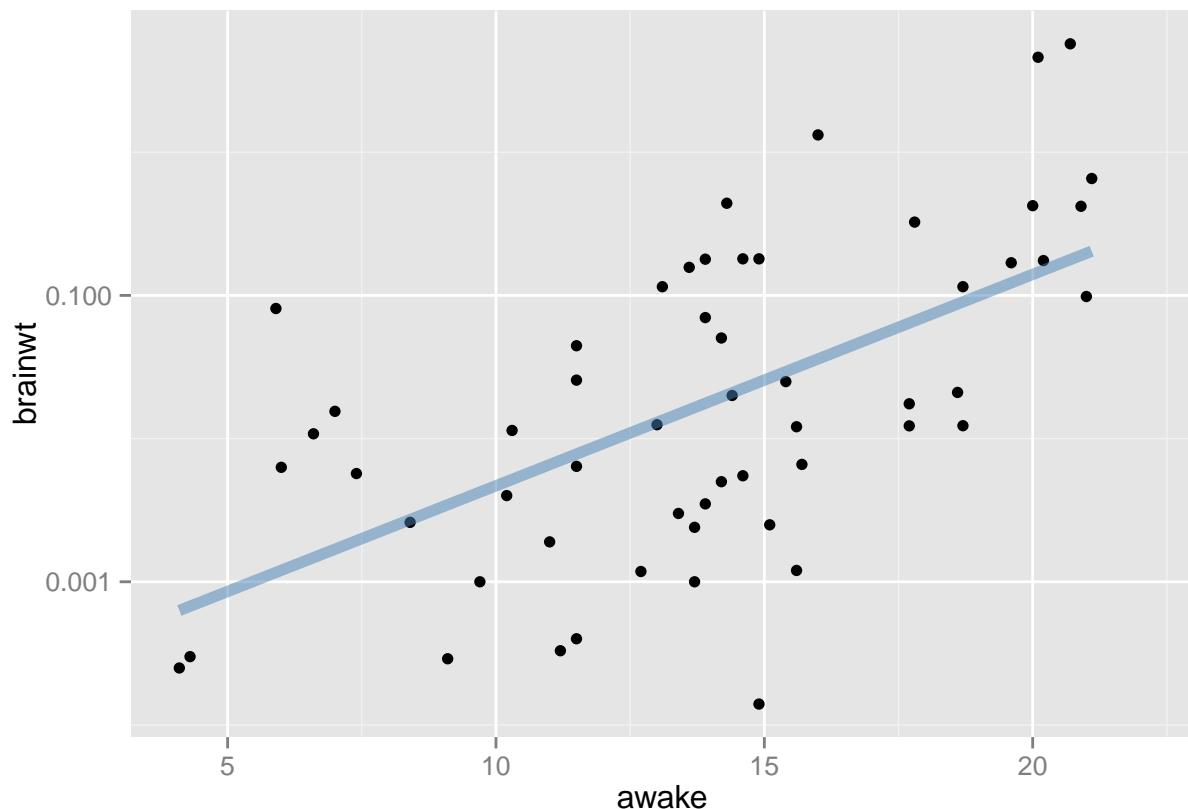
```
## Warning: Removed 22 rows containing missing values (geom_point).
```



```
qplot(awake, brainwt, data = msleep, log = "y") + bestfit
```

```
## Warning: Removed 27 rows containing missing values (stat_smooth).
```

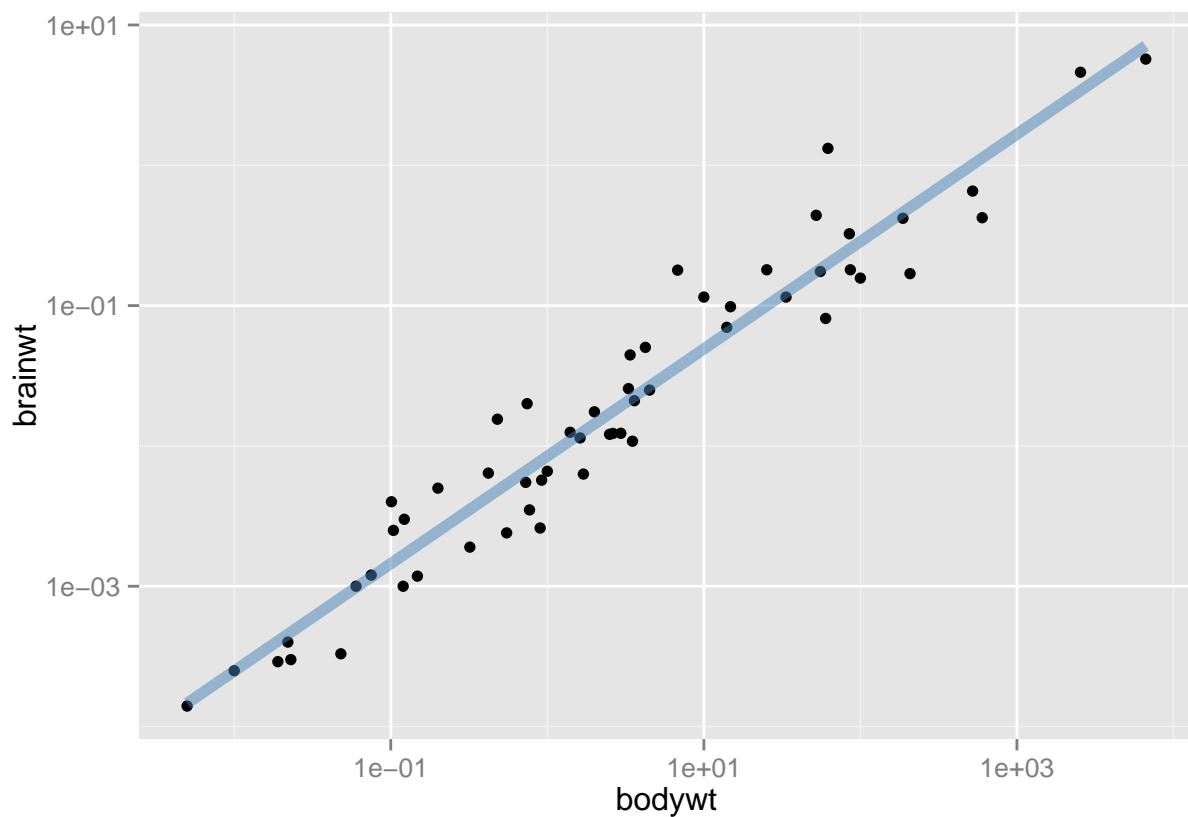
```
## Warning: Removed 27 rows containing missing values (geom_point).
```



```
qplot(bodywt, brainwt, data = msleep, log = "xy") + bestfit
```

```
## Warning: Removed 27 rows containing missing values (stat_smooth).
```

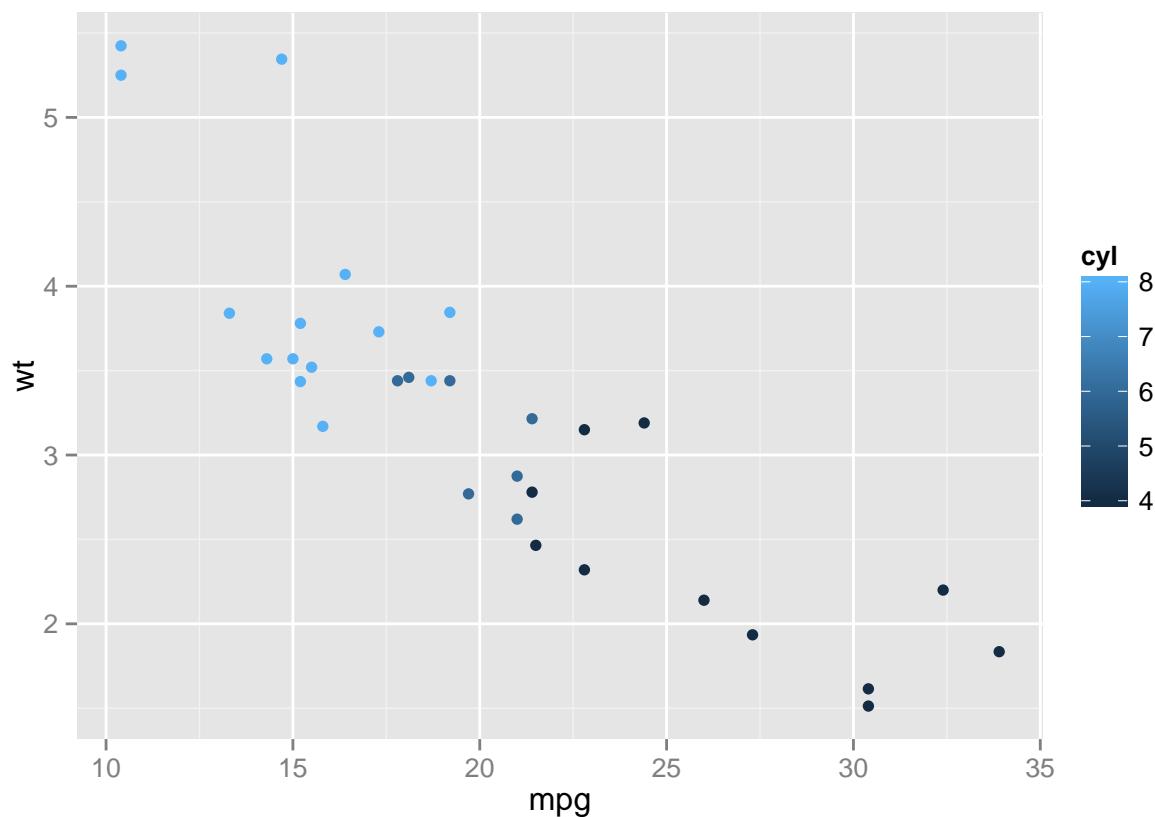
```
## Warning: Removed 27 rows containing missing values (geom_point).
```



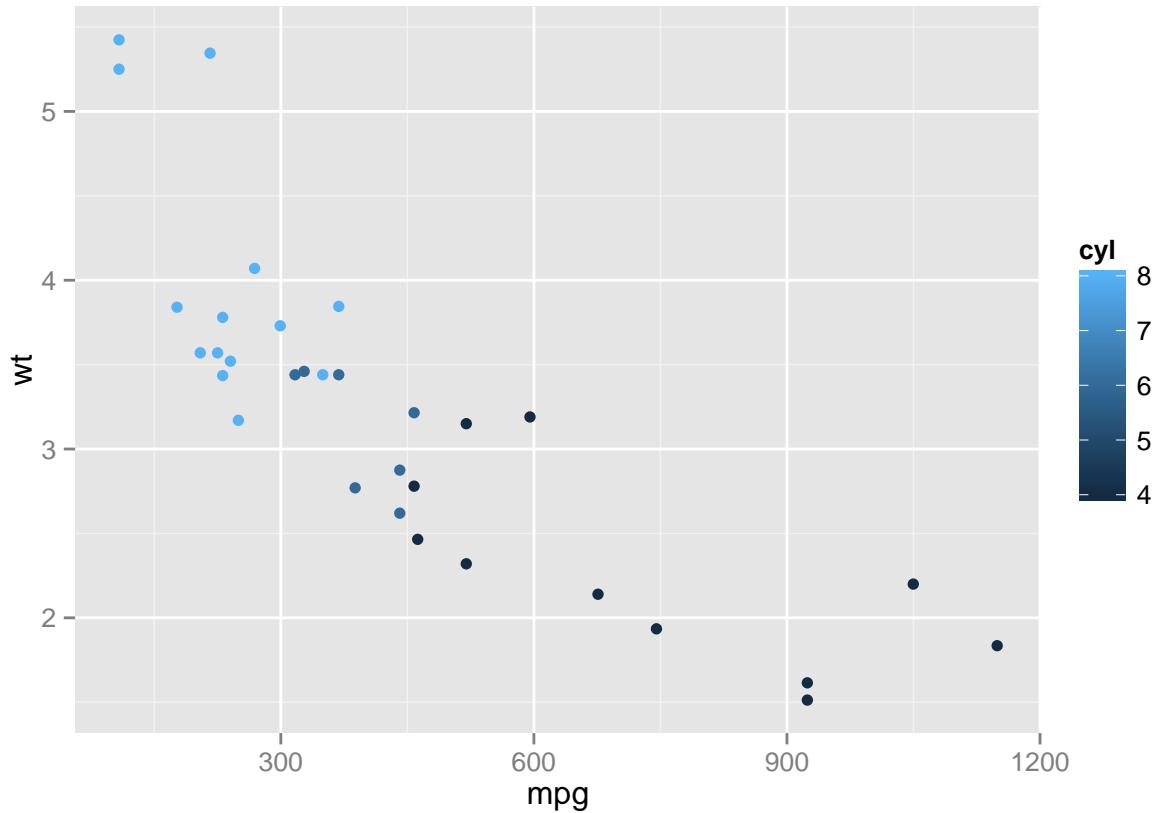
改变图形对象的数据集

使用相同的代码，不同的数据集绘图。“+”表示用添加新的数据集代替原来的数据集。

```
## 用%*% 添加新的数据集来代替原来的数据集
p <- ggplot(mtcars, aes(mpg, wt, colour = cyl)) + geom_point()
p
```



```
mtcars <- transform(mtcars, mpg = mpg^2)
p %+% mtcars
```



数据是以副本的方式而不是以引用的方式存储到图形对象中，这样做的好处是：如果你改变了数据，绘图不会变；其二是，ggplot2 的对象都是自含型，所以可以被存储到磁盘上 save()，并且之后可以被直接加载运行 load()。

图形属性映射

aes() 函数用来将数据变量映射到图形中，从而使变量成为可以被感知的图形属性。aes 里的变量都必须包含于默认的数据集或者图层数据集中，这是保证 ggplot2 对象是自含型的重要方式之一，方便存储和重复使用。

```
## aes 函数的参数
aes(x = weight, y = height, colour = age)
```

```
## List of 3
## $ x      : symbol weight
## $ y      : symbol height
## $ colour: symbol age
```

图和图层

默认的图形属性映射在图形对象初始化时指定，或者过后通过用 + 号修改。

```
# 也可以使用变量的函数值作为参数
aes(weight, height, colour = sqrt(age))
```

```

## List of 3
## $ x      : symbol weight
## $ y      : symbol height
## $ colour: language sqrt(age)

p <- ggplot(mtcars)
summary(p)

## data: mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb
## [32x11]
## facetting: facet_null()

p <- p + aes(wt, hp)
summary(p)

## data: mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb
## [32x11]
## mapping: x = wt, y = hp
## facetting: facet_null()

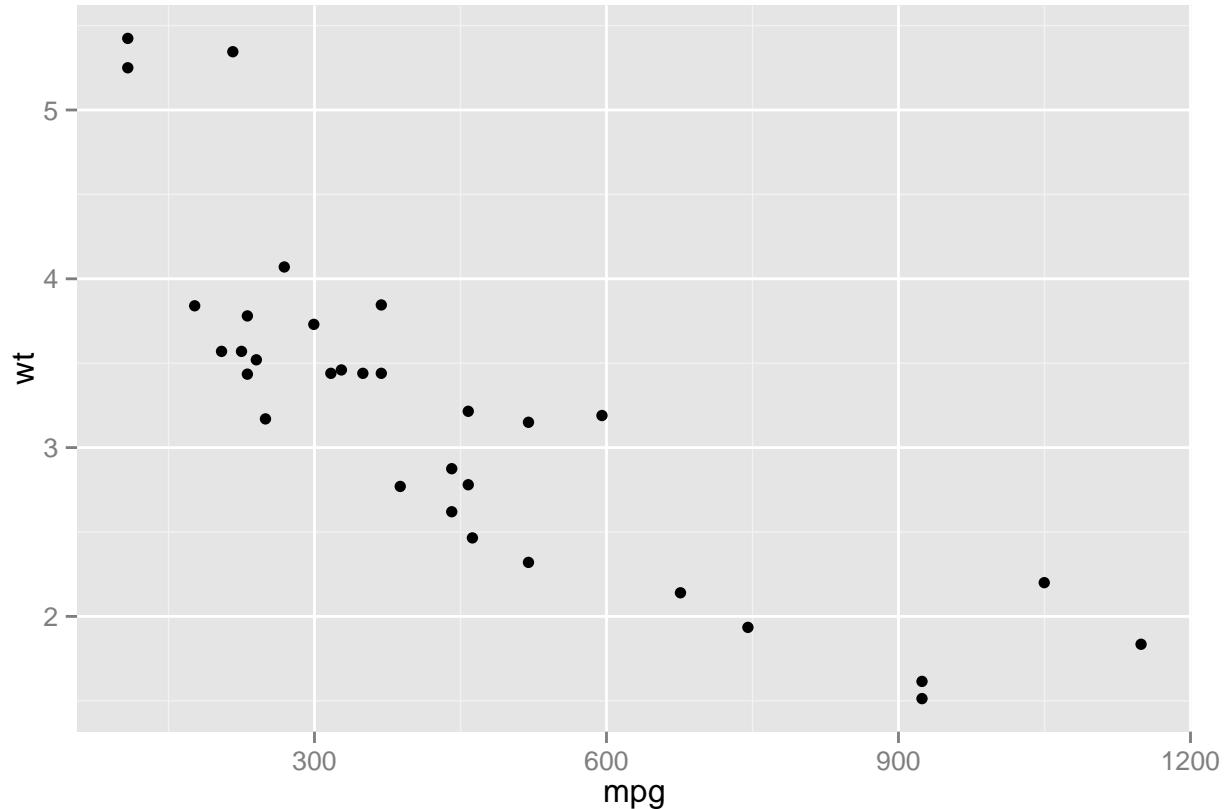
```

再来一个修改添加默认图层映射的例子。

```

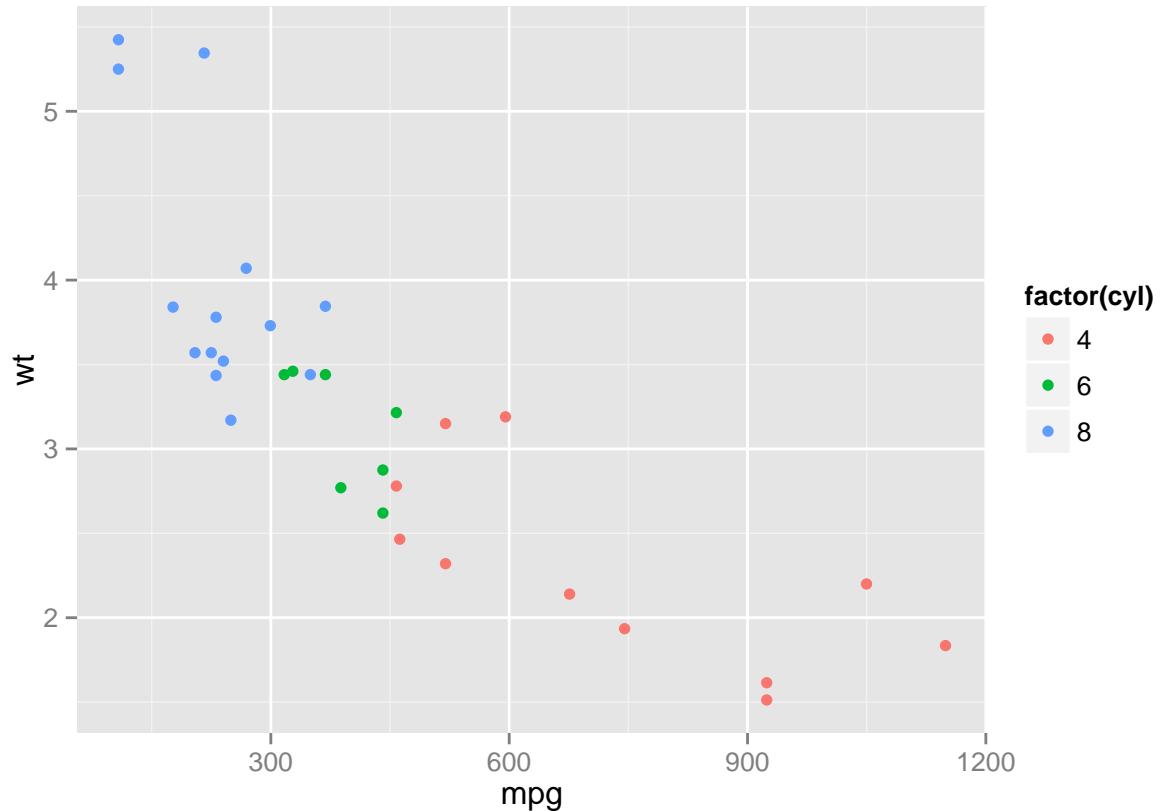
## 使用默认的参数映射来添加图层
p <- ggplot(mtcars, aes(x = mpg, y = wt))
p + geom_point()

```

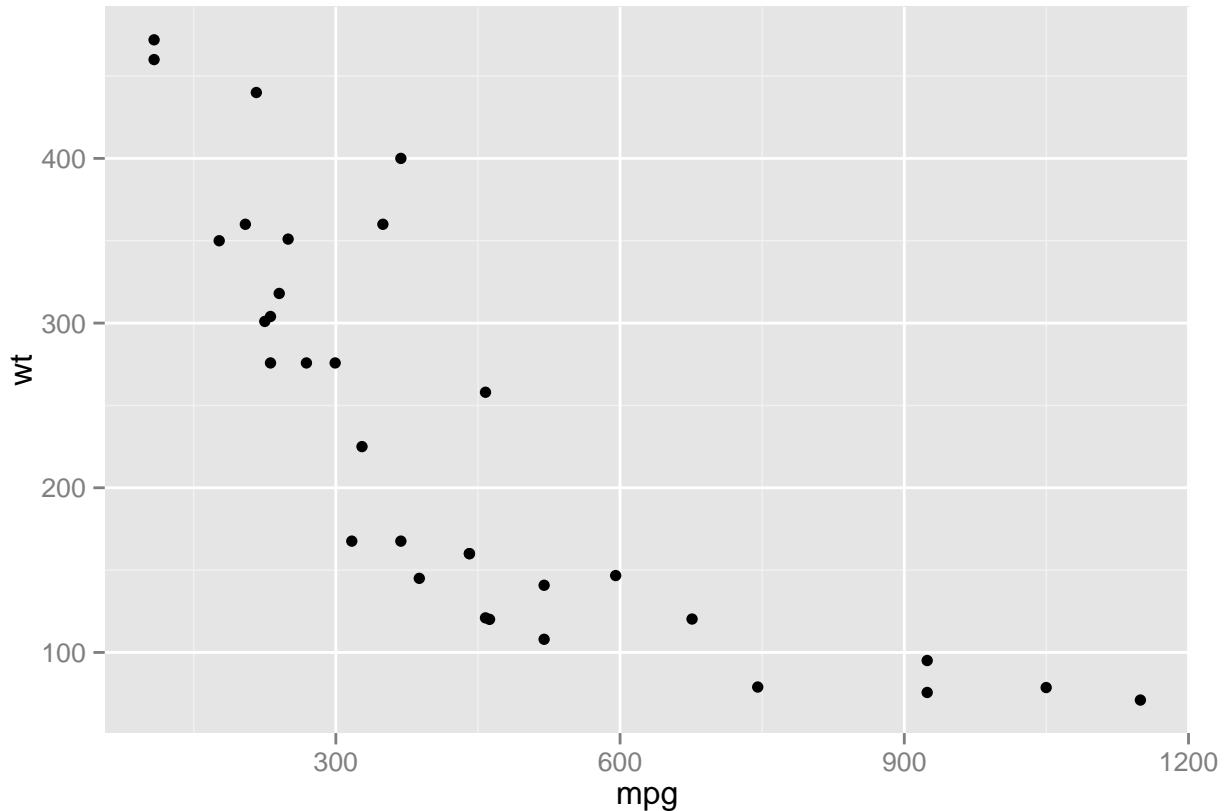


图形对象 p 中默认的映射可以在新图层里进行扩充或修改。

```
## 修改图形属性。用 factor(cyl) 修改颜色 (左), 用 disp 修改 y 坐标 (右)。  
p + geom_point(aes(colour = factor(cyl)))
```



```
p + geom_point(aes(y = disp))
```

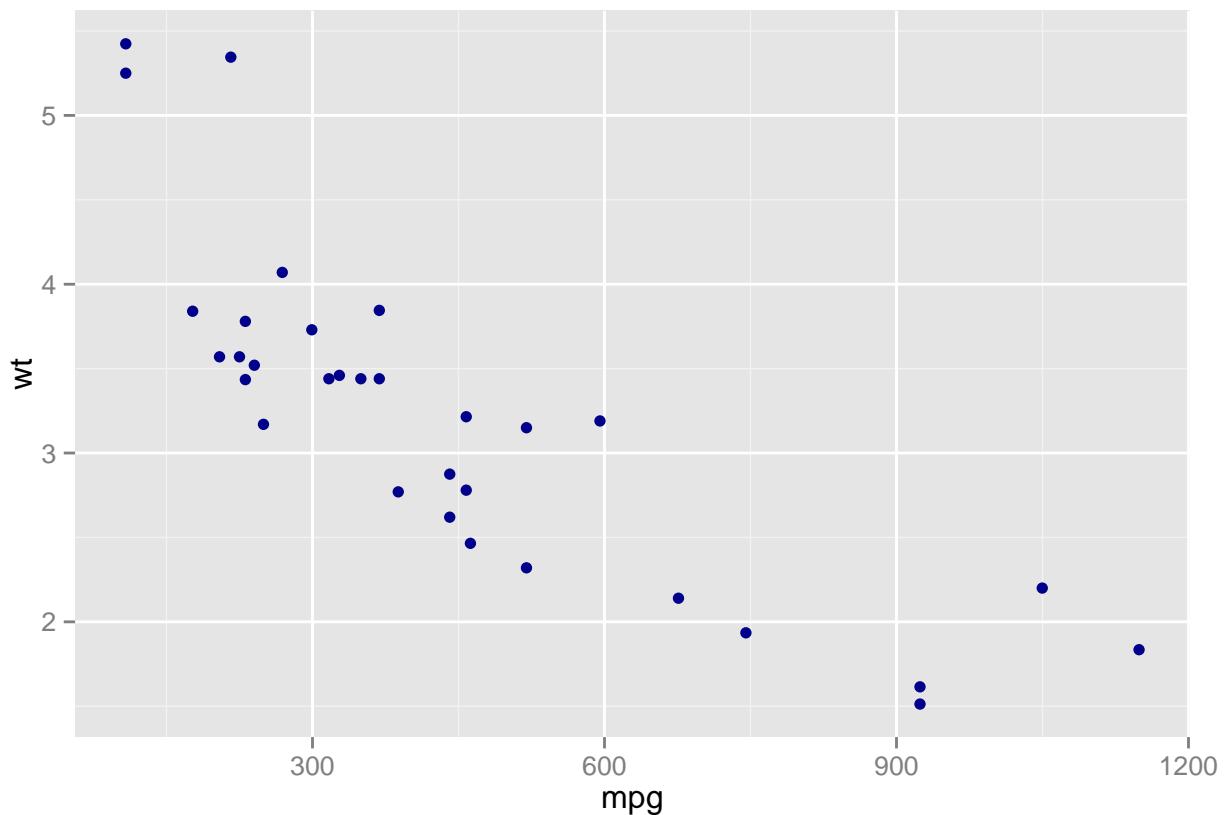


一个图层里设定的图形属性映射只对该图层起作用。

设定和映射

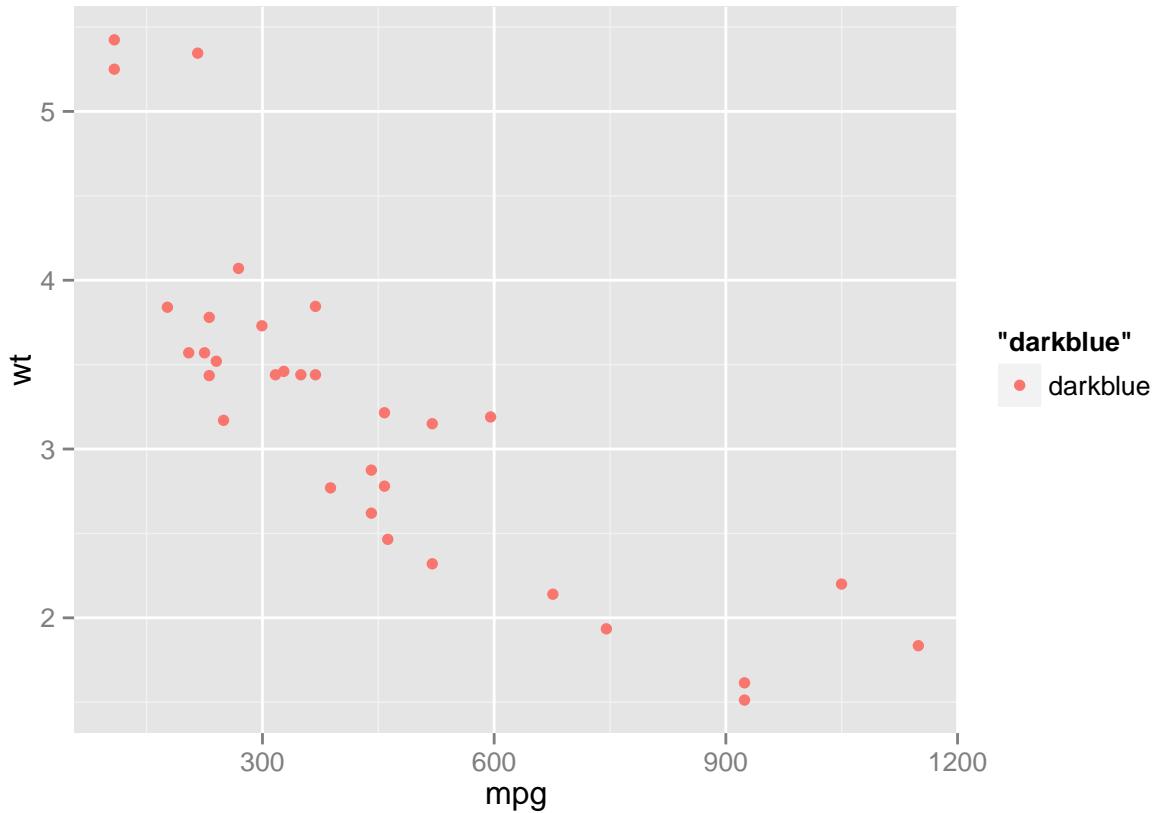
除了可以将一个图形属性映射到一个变量，也可以在图层的参数里将其设定为一个单一值。图形属性可以根据观测的不同而变化，但参数不可以。下面将点的颜色设定为深蓝色。

```
p <- ggplot(mtcars, aes(mpg, wt))  
p + geom_point(colour = "darkblue")
```



下面将 color 映射到 darkblue 颜色，实际上是先创建了一个只含有 darkblue 字符的变量，然后将 color 映射到这个新变量。因为新变量是离散的所有默认的颜色标度将用色轮上等间距的颜色，为桃红色。

```
# 注意这里将颜色映射到'darkblue' 与上面将颜色设定给'darkblue' 的区别  
p + geom_point(aes(colour = "darkblue"))
```



```
## 将颜色设定为'darkblue'(左) 与将颜色映射到'darkblue'(右) 的区别。当颜色映
## 射到'darkblue'时，'darkblue'将被看作一个普通的字符串，使用默认的颜色标
## 度进行标度转换，结果得到了粉红色的点和图例。
```

当颜色映射到 `darkblue` 时，`darkblue` 被看做普通的字符串，使用默认的颜色标度进行标度转换，结果得到粉红色的点和图例。

分组

在 `ggplot2` 中，几何对象大致分为个体几何对象和群组几何对象两大类。个体几何对象对数据框的每一条数据绘制一个可以区别于其他个体的图形对象。相反，群组对象用来表示多条观测，他们可以是某个统计摘要的一个结果，或者是几何对象的基础表示，例如，多边形。线条和路径介于这两者之间：每条线由许多线段组成，而每条线段又代表两个点。

图中所有离散型变量的交互作用被设为分组的默认值，如果没有离散型变量，就需要自定义分组结构，即将 `group` 映射到一个在不同的组有不同取值的变量，单个变量不能正确地分组时，两个变量的组合可以正确分组，用 `interaction()` 函数。

有三种情况是默认分组不能解决的。`nlme` 包里的一个简单的纵向数据集 `Oxboys`，记录了 26 名男孩 `Subject`，在 9 个不同时期 `Occasion` 所测定的身高和中性化后的年龄。

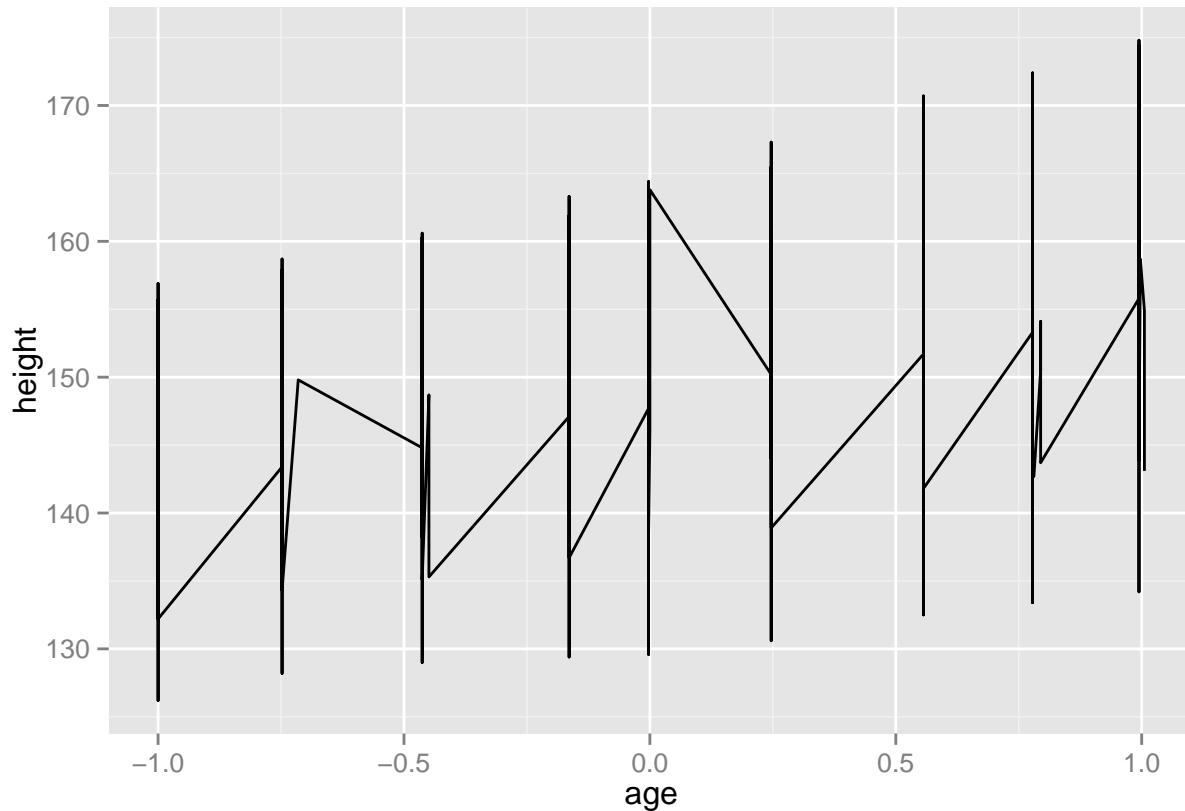
多个分组和单个图形属性

将数据分成若干组，并用相同的方式对每个组进行渲染。下面是一个时间序列图，每条线代表一个男孩。你可以看到每个男孩的成长轨迹，但是不能识别哪个男孩是哪条轨迹。

```

## 图 4.3 正确分组时 (分组变量 group =
## Subject) 每个个体的折线图 (左)。错误的分组时连
## 接所有观测点的折线图 (右)。此处省略了分组图形属性, 效果等同于 group =
## 1。
data(Oxboys, package = "nlme")
# 左图的代码
p <- ggplot(Oxboys, aes(age, height, group = Subject)) + geom_line()
# 或
# qplot(age, height, data = Oxboys, group = Subject, geom = "line")
# 右图的代码
qplot(age, height, data = Oxboys, geom = "line")

```



第二个图因为没有正确分组得到没有意义的折线图。

不同图层上的不同分组

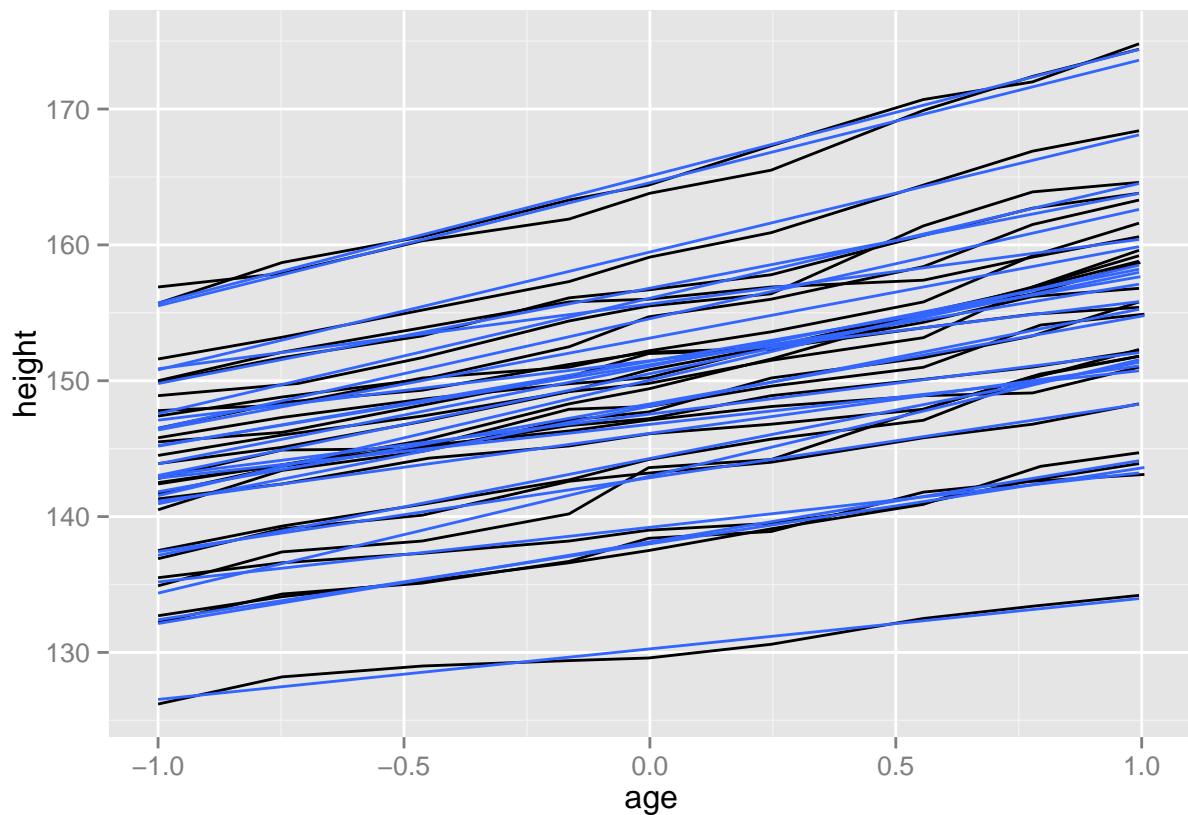
有时我们想根据不同水平下的数据整合来对统计汇总信息进行图形绘制, 从而不同的图层可能有不同的分组图形属性, 因此, 有的图层展示个体水平数据, 有的图层则展示更大组群的统计信息。

根据所有男孩的年龄和身高在图中添加一条光滑线条。

```

## 给 Oxboys 数据添加光滑曲线。左图用了和折线图同样的分组变量, 得到了每个男
## 孩的拟合直线。右图在平滑层里用了 aes(group = 1), 得到了所有男孩的拟合直
## 线。 左图
p + geom_smooth(aes(group = Subject), method = "lm", se = F)

```



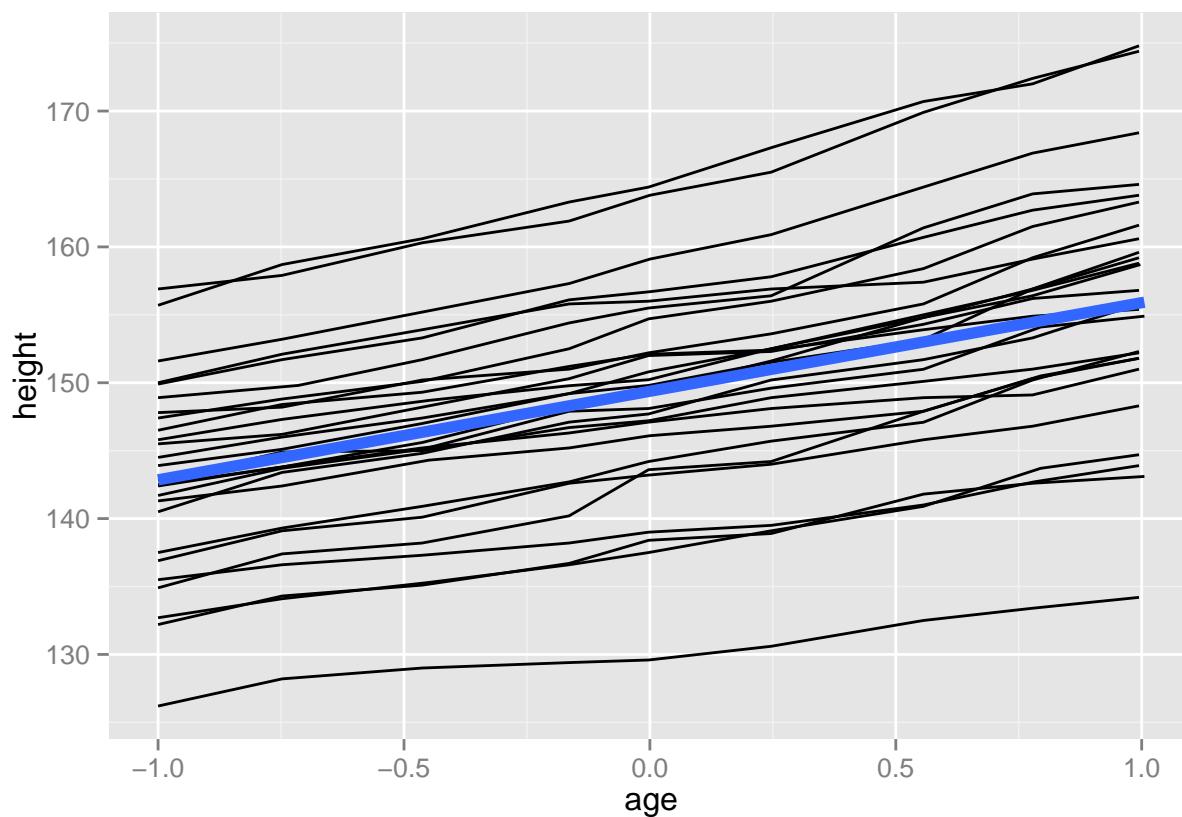
```
# 或
```

```
# qplot(age, height, data = Oxboys, group = Subject, geom = "line") + geom_smooth(method = lm)
```

这并不是想要的结果，无意间给每一个男孩添加额、了一条光滑线条。因此，新图层需要一个不同的分组图形属性，group=1，这样所绘出的线条才是基于整体数据的。修改后的代码如下：

```
# 右图
```

```
p + geom_smooth(aes(group = 1), method = "lm", size = 2, se = F)
```

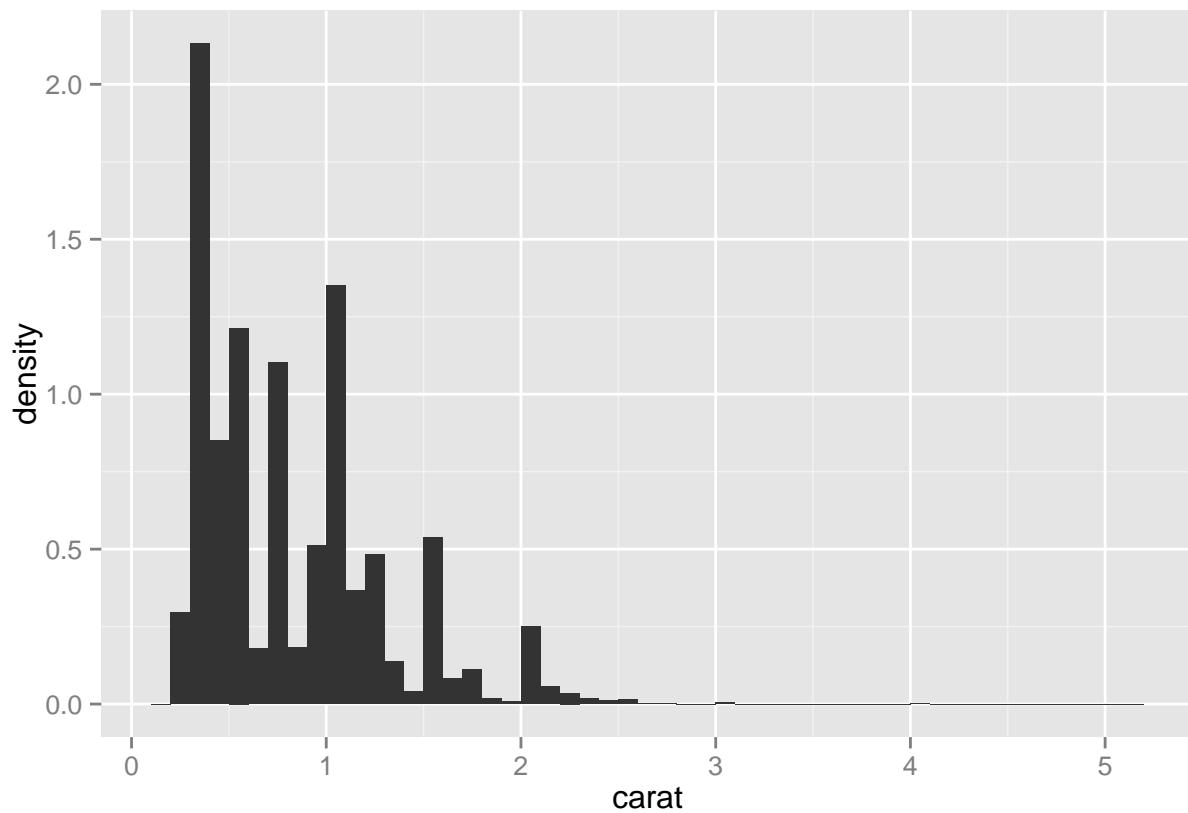


```
# 或
# qplot(age, height, data = Oxboys, group = Subject, geom = "line") + geom_smooth(aes(group = 1))
```

统计变换

以某种方式对数据信息进行汇总。count，每个组里观测值的数目。density 每个组里观测值的密度。x，组的中心位置。这些生成变量可以随时被直接调用。下面给出钻石数据集中的克拉的密度直方图。

```
## 例：生成变量
ggplot(diamonds, aes(carat)) + geom_histogram(aes(y = ..density..), binwidth = 0.1)
```



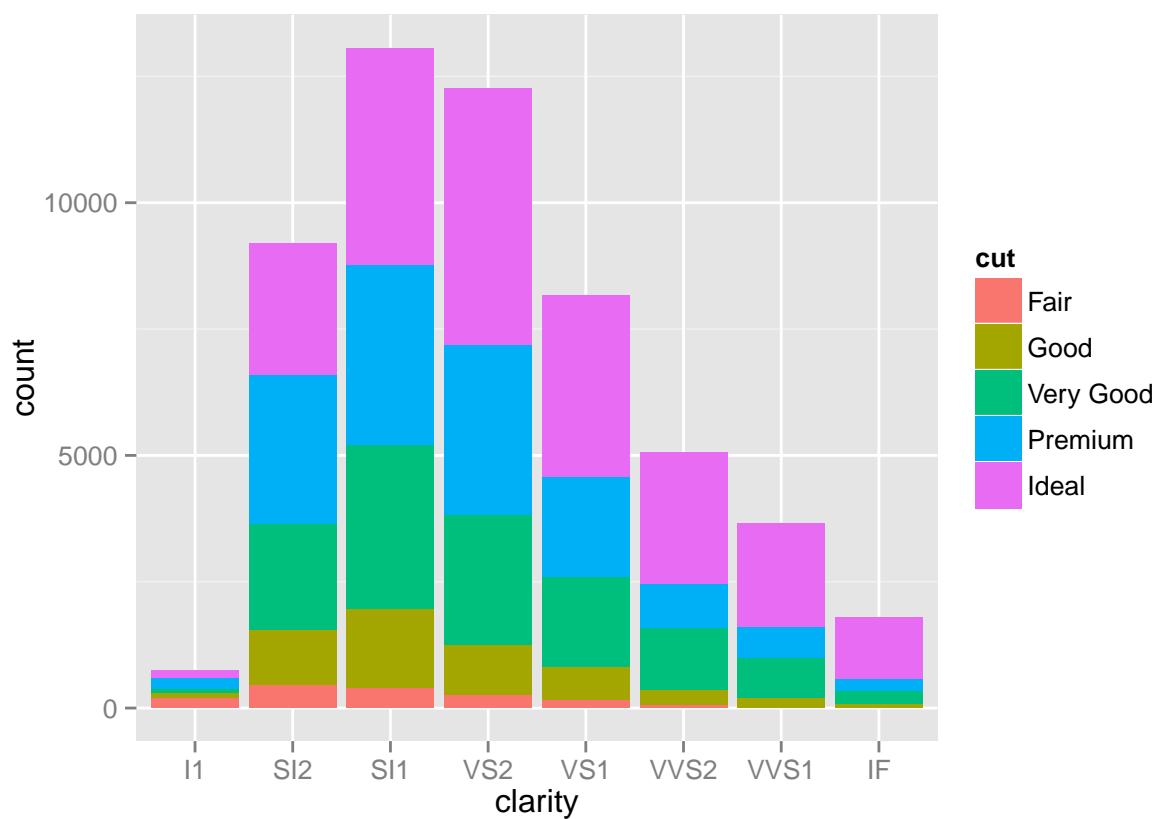
```
# 或
# qplot(carat, ..density.., data = diamonds, geom = "histogram", binwidth = 0.1)
```

生成变量的名字必须用..围起来，防止原数据集中的变量和生成变量重名时造成混淆。

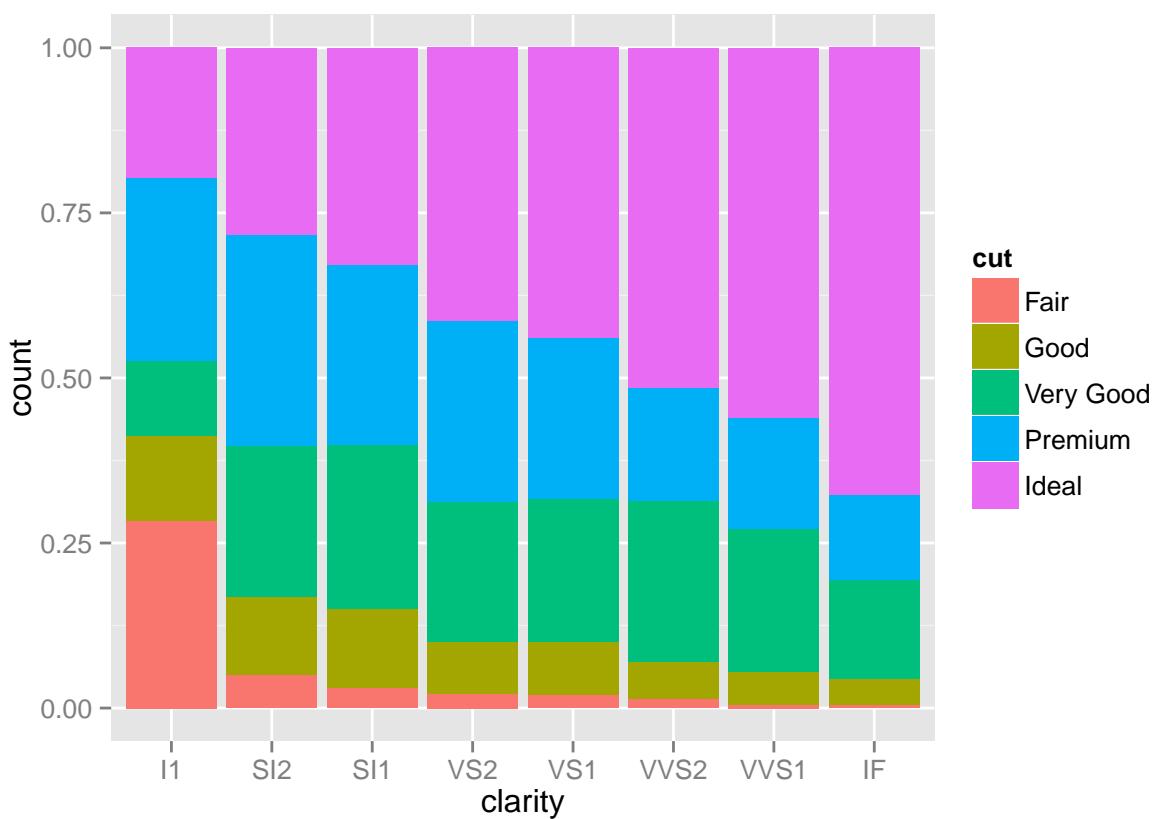
位置调整

对层中的元素进行微调，防止重叠。

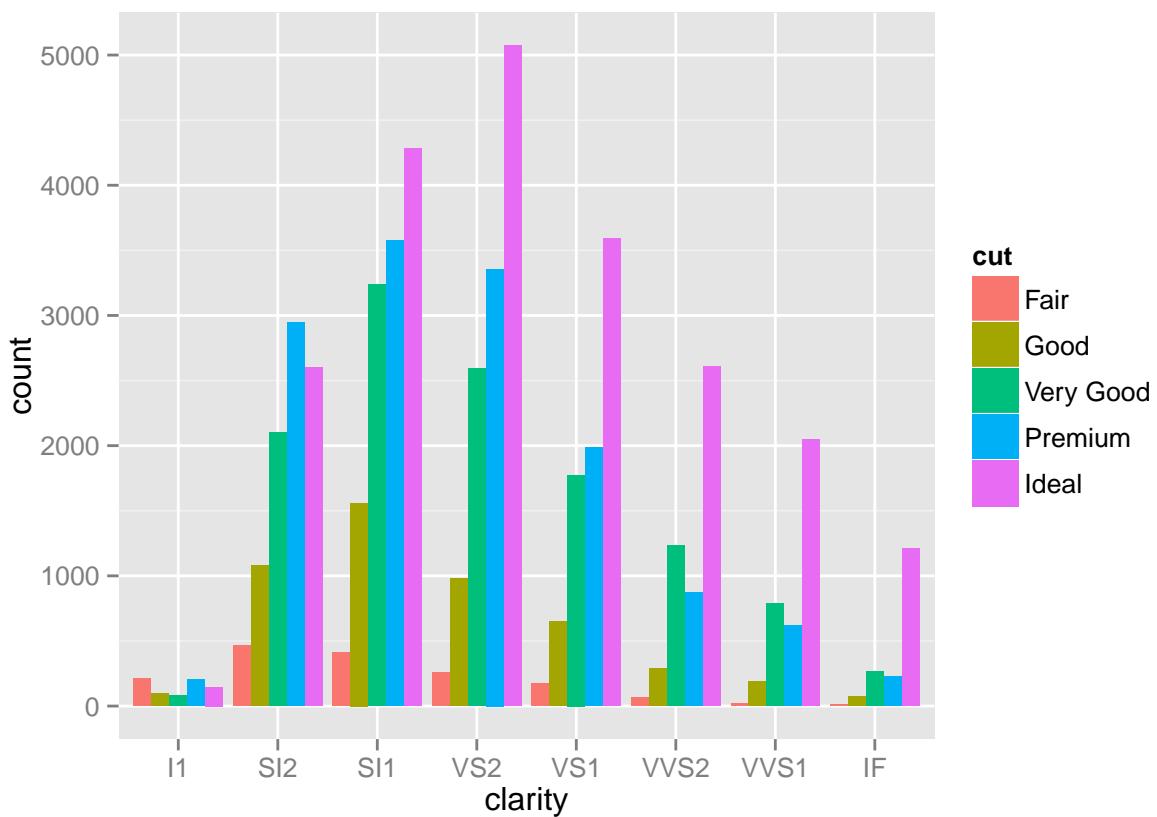
```
## 应用于条形图的三种位置调整。从左到右依次是：堆叠（stacking），填充
## (filling) 和并列 (dodging)
dplot <- ggplot(diamonds, aes(clarity, fill = cut))
dplot + geom_bar(position = "stack")
```



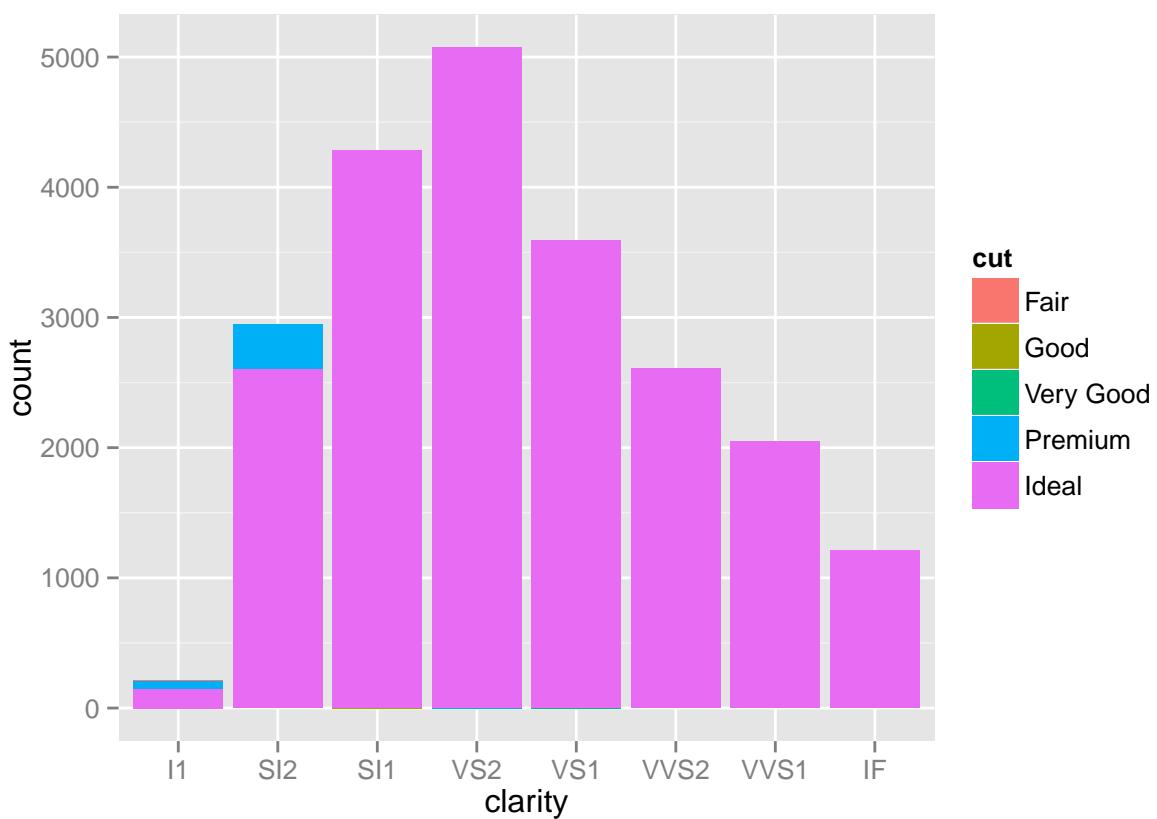
```
dplot + geom_bar(position = "fill")
```



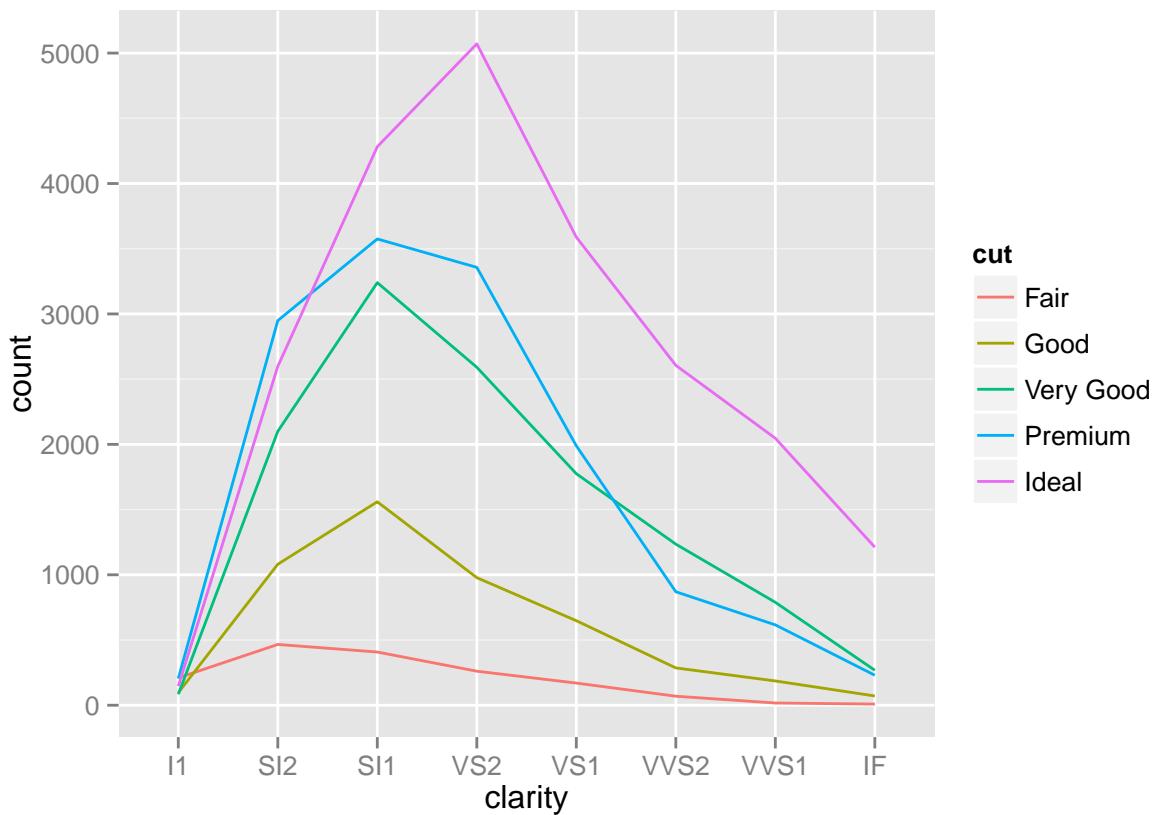
```
dplot + geom_bar(position = "dodge")
```



```
## 同一调整 (identity
## adjustment) 不适用于条形图 (左), 因为后画的条形会挡住先
## 画的条形。但它适用于线型图 (右), 因为线条不存在相互遮掩的问题。
dplot + geom_bar(position = "identity")
```



```
qplot(clarity, data = diamonds, geom = "line", colour = cut, stat = "bin",
      group = cut)
```

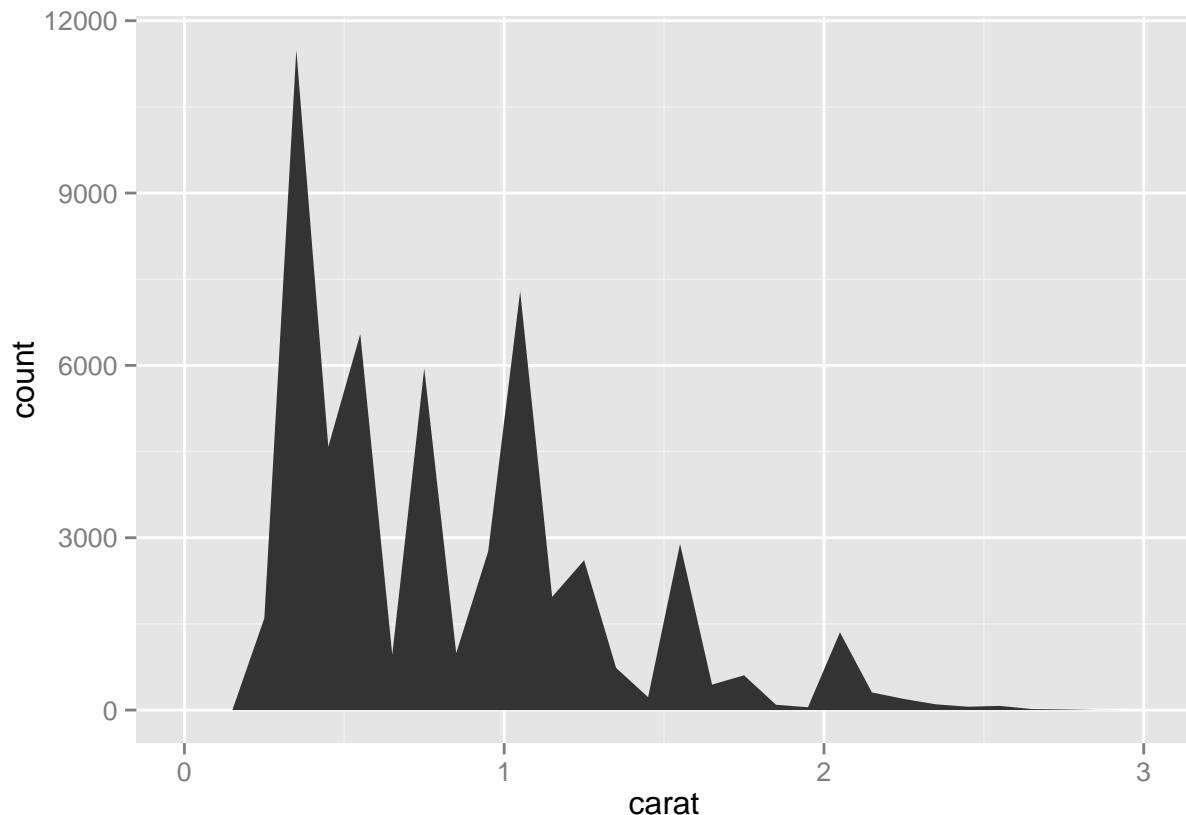


整合

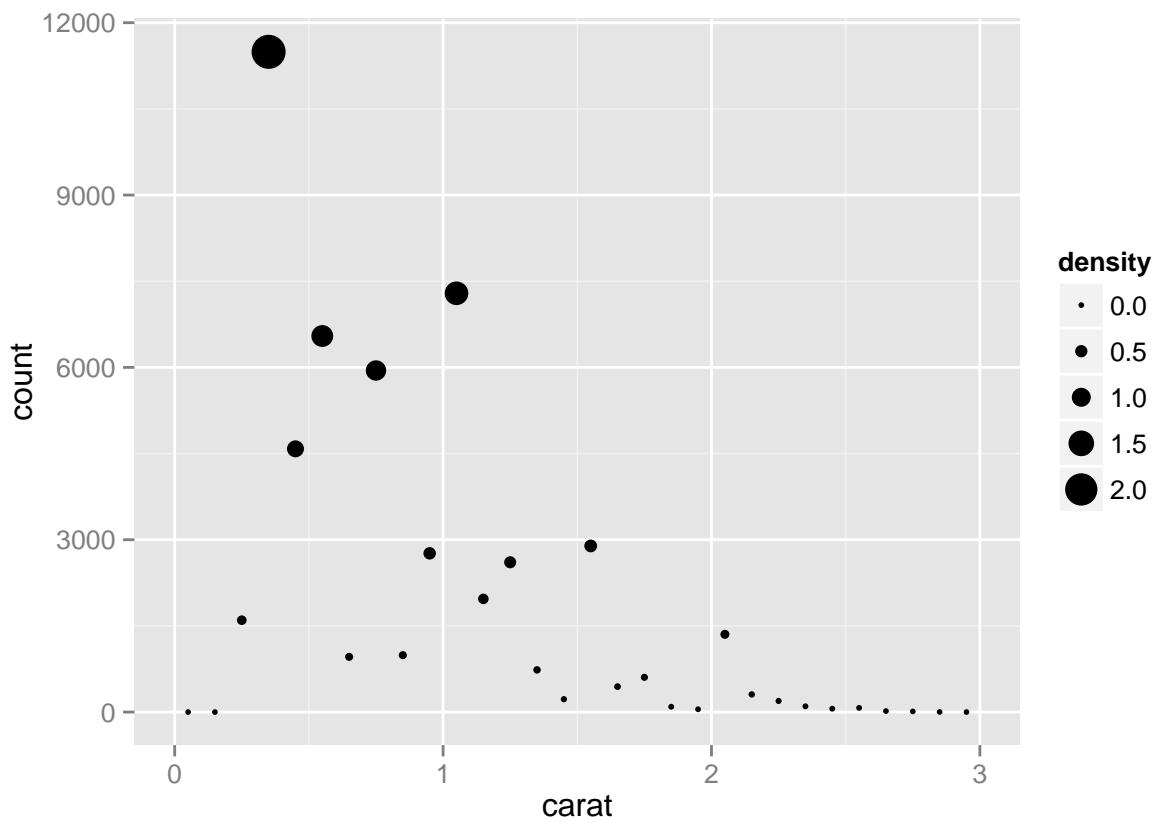
结合几何对象和统计变换

下图给出了直方图的三个变体，都是基于直方图的统计变换，但使用了不同的几何对象来展示结果：面积，点和瓦块。

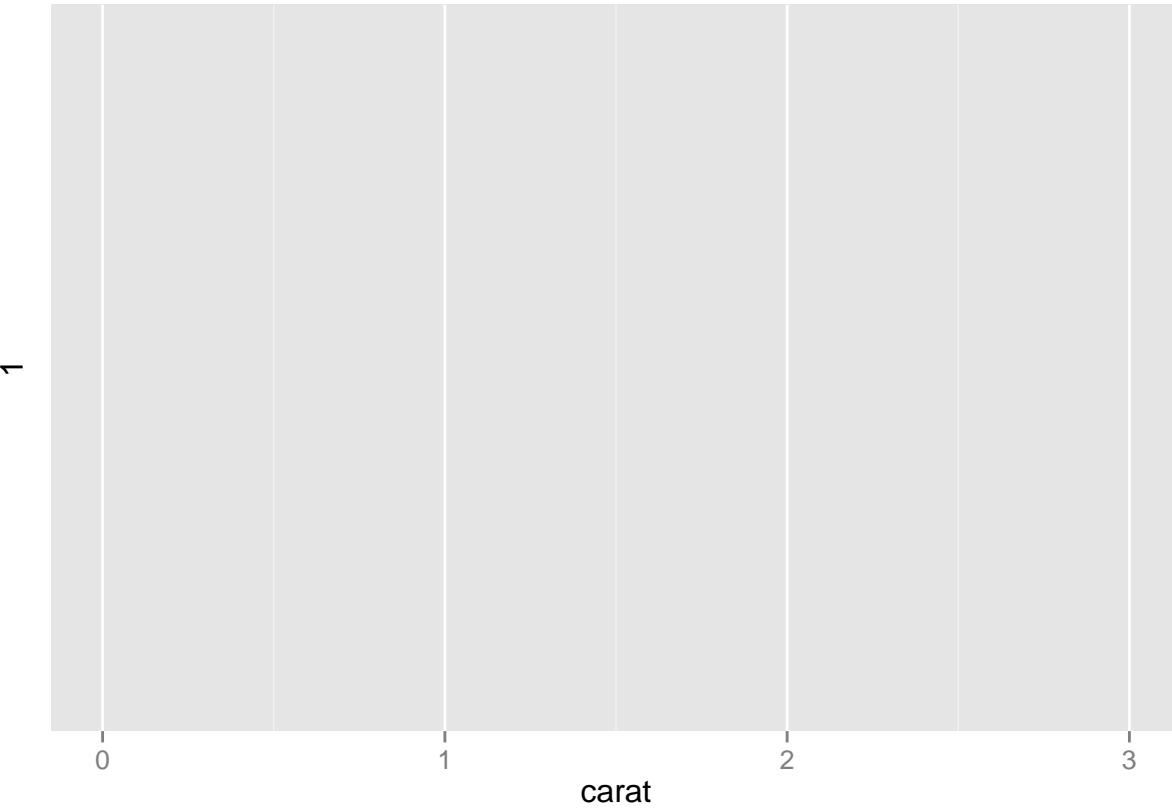
```
## 图 4.10 直方图的三种变体。频率多边形 (frequency
## polygon) (左); 散点图, 点的大小和
## 高度都映射给了频率 (中); 热图 (heatmap) 用颜色来表示频率。
d <- ggplot(diamonds, aes(carat)) + xlim(0, 3)
d + stat_bin(aes(ymax = ..count..), binwidth = 0.1, geom = "area")
```



```
d + stat_bin(aes(size = ..density..), binwidth = 0.1, geom = "point", position = "identity")  
## Warning: Removed 2 rows containing missing values (geom_point).
```



```
d + stat_bin(aes(y = 1, fill = ..count..), binwidth = 0.1, geom = "tile", position = "identity")
```



改变图形属性和数据集

将不同的数据画在不同的图层上，把不同的数据画在同一个图上有什么作用呢？比如用拟合模型得出的预测值来扩充原数据集。

前面给出了每个男孩的线性拟合曲线和所有男孩的线性拟合曲线（简称群体模型），两个模型都不完美，群体模型忽视个体内的相关性，个体模型没能利用一般增长模型的信息，从而不能精确预测个体，实际上可用混合模型来改进。

```
## 例: nlme 包的 Oxboys 数据集
require(nlme, quiet = TRUE, warn.conflicts = FALSE)
model <- lme(height ~ age, data = Oxboys, random = ~1 + age | Subject)
oplot <- ggplot(Oxboys, aes(age, height, group = Subject)) + geom_line()
```

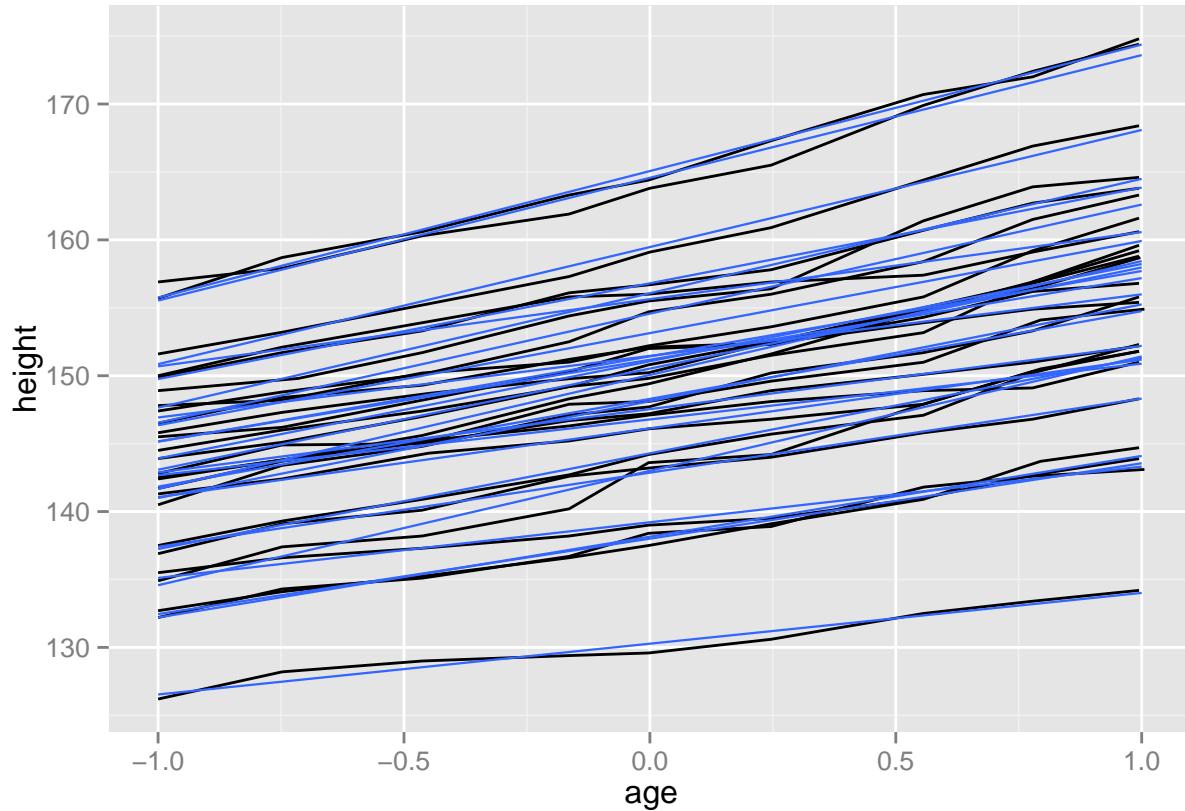
随后，对预测的生长轨迹和实际的生长轨迹进行对比，建立一个包含所有年龄和个体组合的网格数据框。接下来把模型的预测值添加到刚刚生成的数据集中，变量名叫 height。

```
age_grid <- seq(-1, 1, length = 10)
subjects <- unique(Oxboys$Subject)

preds <- expand.grid(age = age_grid, Subject = subjects)
preds$height <- predict(model, preds)
```

得到预测值后，把它和原始数据绘制在同一张图上。因为在新的数据集 preds 里，我们使用了与原始数据 Oxboys 相同的变量名，并且想使用相同的分组图形属性，所以不用修改任何图形属性，只需修改默认的数据集即可。设定颜色和大小两个图形属性参数便于比较图形。

```
oplot + geom_line(data = preds, colour = "#3366FF", size = 0.4)
```

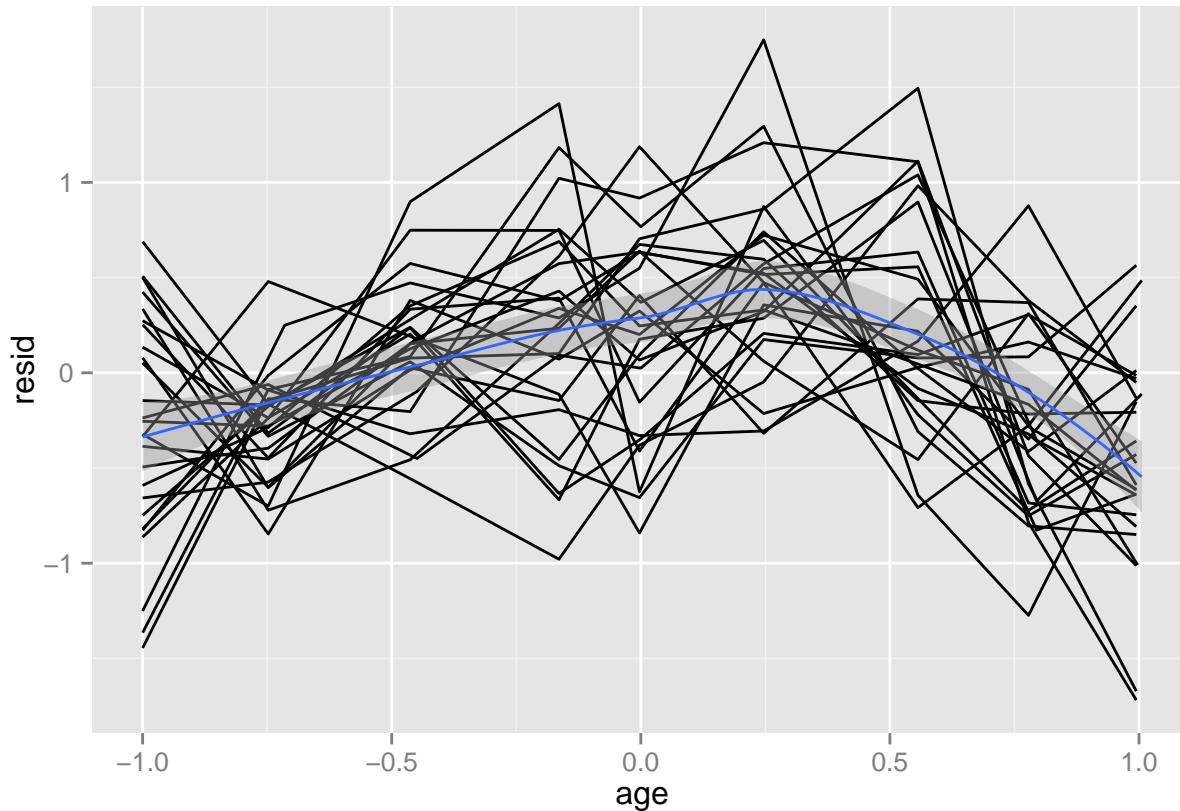


进一步改进，把拟合值 fitted，残差 resid 都添加到原始数据里去，然后更新数据集%+%，将默认的 y 图形属性改为 resid，最后对整个数据添加一条光滑曲线。

```
Oxboys$fitted <- predict(model)
Oxboys$resid <- with(Oxboys, fitted - height)

oplot %+%
  Oxboys +
  aes(y = resid) +
  geom_smooth(aes(group = 1))

## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'method'
```

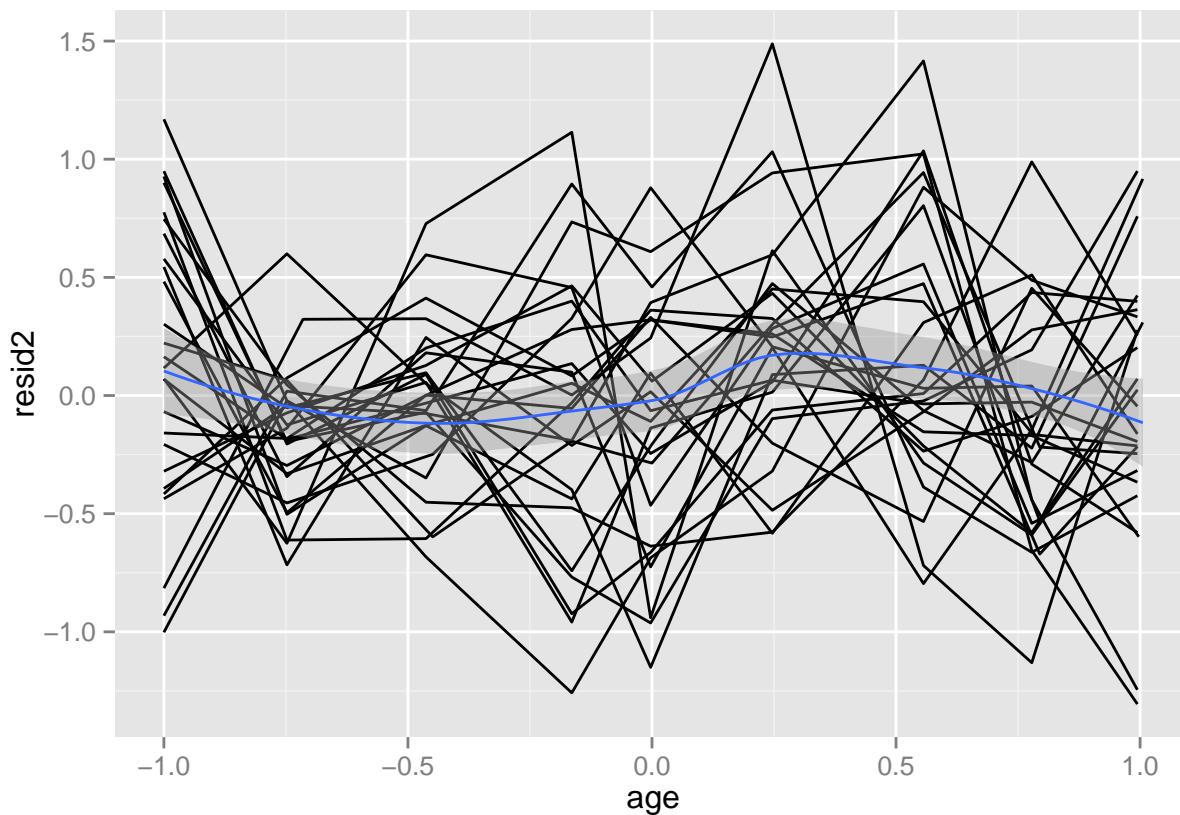


从图中看到残差不是随机分布的，因此建立的模型有缺陷，向模型中添加一个二次项，再次计算拟合值与残差并重新绘制残差图。

```
model2 <- update(model, height ~ age + I(age^2))
Oxboys$fitted2 <- predict(model2)
Oxboys$resid2 <- with(Oxboys, fitted2 - height)
```

```
oplot %+%
  Oxboys +
  aes(y = resid2) +
  geom_smooth(aes(group = 1))
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess. Use 'me
```



对图形的修改非常简单，这正是 **ggplot2** 图层功能所秉承的理念，使得反复拟合和评估模型变得轻松自然。

关于 R 的 **markdown** 生成 **pdf** 文件

RStudio 是 R 语言的一款开源的 GUI 软件，可以让“用不起 Matlab 只好用 R”的用户体验到很像 Matlab 一样的开发环境，包括变量的显示、package 的基本操作、帮助文档等一系列图形界面。RStudio 另一个隐藏功能是可以当做 Markdown 编辑器来用（R Markdown），可以根据数据处理结果快速生成报告文档，这一功能主要通过 Package Knit 及相关组件完成。R Markdown 的两大特别之处，一是通过 Pandoc 将 Markdown 转化成 LaTex，再由强大的 LaTex 转换成 HTML、PDF、Word，理论上来说借助 LaTex 可以生成学术论文、期刊杂志、数据报告等规范格式的文档；另一大特点是整合了 R 语言的环境，可以在 Markdown 语法中 code block 直接执行 R 语言代码并将结果插入文档。安装 package rmarkdown：可以通过 RStudio 中的新建按钮创建 R Markdown 文件，此时可能提示安装 rmarkdown 包：

```
install.packages(rmarkdown)
```

- 创建文档可以选择文档标题、作者以及将要输出的文档格式等，这些也可以在之后更改：第一次按 Knit PDF（或 Knit HTML）时可能出现错误：

```
Knit PDF : pandoc document conversion failed with error 43
```

- 可以通过安装 github 上最新的版本解决：

```
install.packages("devtools") # 如果以前没有安装 devtools 包
devtools:::install_github("rstudio/rmarkdown")
```

- 第二个问题是当文档中有中文的情况，可能提示：

```
! Package inputenc Error: Unicode char \u8: 年 not set up for use with LaTeX.
Try running pandoc with --latex-engine=xelatex.
```

- 如果有中文，LaTeX 引擎需要选择为 xelatex，可以通过下图的方式进行更改：

或者直接在文档头信息中加入：

```
output:
  pdf_document:
    latex_engine: xelatex
```

这样就可以正常输出为 PDF 文档了，但是会发现所有的中文全部都是空白，这个主要是 LaTeX 的配置问题，中文需要中文字体来渲染，可以通过在文件头中引入 LaTeX 文件进行配置：

```
outputs:
  pdf_document:
    includes:
      in_header: header.tex
    latex_engine: xelatex
```

其中 header.tex 可以是：

```
\usepackage{xeCJK}
\setCJKmainfont{楷体} % 字体可以更换
\setmainfont{Georgia} % 设定英文字型
\setromanfont{Georgia} % 字型
\setmonofont{Courier New}
```

总结下，看起来复杂，其实过程是：编辑好的 Rmd 文件 -> md 文件 -> pdf 文件。还存在一个问题就是，Rmd 文件中有中文时，转换 pdf 文件会出错（已经是 utf-8）的编码了，这个还要好好研究下。另外 Rmd 文件的文件名不能为中文，否则 knitr 生成 html 文件时就会报错。

关于 R 代码块的样式设计请参考帮助 chunk.

错误解决方案请参考：[\(http://blog.rainy.im/2015/05/16/rmarkdown-in-rstudio/\)](http://blog.rainy.im/2015/05/16/rmarkdown-in-rstudio/) 或者论坛：[\(http://cos.name/cn/\)](http://cos.name/cn/).

参考文献

- [美] 哈德利. 威科姆, 统计之都译, 西安交大出版社, ggplot2: 数据分析与图形艺术。
- 蒋家坤等.R 语言实用数据分析和可视化技术. 机械工业出版社
- 陈开江译. 机器学习实用案例解析. 机械工业出版社.201304.
- 吴喜之. 应用时间序列分析-R 软件陪同. 人民大学出版社
- 王亮等译. R 语言的科学编程与仿真. 西北工业大学出版社
- 方匡南, 朱建平, 姜叶飞.R 数据分析——方法与案例详解（双色）. 电子工业出版社,201502.
- 王斌会. 多元统计分析及 R 语言建模. 暨南大学出版社.201405.
- 胡伟. LATEX2ε 完全学习手册. 清华大学出版社, 201101.