

R 语言绘图速查手册 *v0.1beta*

俞丽佳

2016-05-20

目录

引言	6
版权声明	6
参考手册使用方法	6
基本绘图命令和 <i>ggplot2</i> 包简介	6
散点图	8
<i>graphics::points()</i> 基本用法	8
<i>ggplot2::geom_point()</i> 基本用法	8
折线图	9
<i>graphics::lines()</i> 基本用法	10
<i>ggplot2::geom_line()</i> 基本用法	10
条形图	11
<i>graphics::barplot()</i> 基本用法	11
<i>barplot()</i> 堆栈式条形图	11
<i>graphics::barplot()</i> 依次排列的条形图	12
<i>ggplot2::geom_bar()</i> 基本用法	13
数据格式转换	14
<i>ggplot2::geom_bar()</i> 堆栈式条形图	14
<i>ggplot2::geom_bar()</i> 依次排列的条形图	15
<i>ggplot2::geom_bar()</i> 比例式条形图	15
面积图	16
<i>graphics::polygon()</i> 基本用法	16
<i>ggplot2::geom_area()</i> 基本用法	17
密度估计图	18
用 <i>graphics</i> 画密度估计图	18
<i>ggplot2::geom_density()</i>	19
频率图像	19
用 <i>graphics</i> 包画频率图像	19
用 <i>ggplot2::geom_freqpoly()</i> 画频率图像	20

- 直方图 20
 - `graphics::hist()` 基本用法 21
 - `ggplot2::geom_hist()` 基本用法 21
- 箱线图 22
 - `graphics::boxplot()` 基本用法 22
 - `graphics::geom_boxplot()` 基本用法 23
 - 用 `graphics::stat_boxplot()` 为箱线图添加 `error bar` 23
- 提琴图 24
 - `vioplot::vioplot()` 基本用法 24
 - `ggplot2::geom_violin()` 基本用法 25
 - 在提琴图中添加箱线图 26
 - 在提琴图中添加均值和标准差信息 26
- Cleveland 点图 26
 - `graphics::dotchart()` 27
 - `ggplot2::geom_dotplot()` 27
- 热图 28
 - `graphics::heatmap()` 基本用法 28
 - `ggplot2::geom_tile()` 基本用法 28
 - `pheatmap::pheatmap()` 基本用法 29
- PCA 第一第二主成分平面图 31
 - `ggbiplot::ggbiplot()` 基本用法 31
- 层次聚类图 31
 - `plot.hclust()` 基本用法 31
 - 用 `ggdendro::dendrograms` 包画层次聚类图 32
 - 用 `ape` 包绘制系统发育树 33
- 关于绘图，我们还关心的其他问题 34
 - 如何添加 `title`，`x` 轴和 `y` 轴标签 34
 - 如何在一个画布里放多个图片 35
 - 如何旋转条形图坐标轴 39
 - `ggplot2` 的背景怎么去掉 39

图目录

1	基本散点图；200 个正态分布的随机数	8
2	point 散点图；200 个正态分布的随机数	9
3	geom_point 散点图；200 个正态分布的随机数	9
4	基本折线图；10 个正态分布的随机数	10
5	lines() 折线图；刹车速度与滑行距离的关系	10
6	geom_line() 连接观测值；美国人口失业情况折线图	11
7	barplot() 基本条形图；统计泊松分布随机数	11
8	barplot() 堆栈式条形图；分年龄的人口信息被叠加在一起	12
9	barplot() 按分类依次排列的条形图	13
10	geom_bar() 基本条形图	13
11	geom_bar() 堆栈式条形图	14
12	geom_bar() 依次排列式条形图	15
13	geom_bar() 比列式条形图	15
14	polygon() 密度图	16
15	polygon() 面积堆积图	17
16	geom_area() 堆积面积图	18
17	密度估计图	18
18	两个核密度估计图	19
19	geom_density() 核密度估计图	19
20	用 Graphics 函数画频率图	20
21	geom_freqpoly() 频率图	20
22	hist() 直方图	21
23	geom_hist() 直方图	22
24	boxplot() 箱线图	23
25	geom_boxplot() 箱线图	23
26	错误的 error bar 箱线图	24
27	带 error bar 的箱线图	24
28	vioplot() 提琴图	25
29	geom_violin() 提琴图	25
30	添加箱线图信息的提琴图	26
31	添加均值和标准差信息的提琴图	26
32	dotchart() 绘制 Cleveland 点图	27
33	geom_dotplot() 绘制 Cleveland 点图	28
34	用 heatmap() 绘制热图	28
35	geom_tile() 绘制热图	29
36	pheatmap() 绘制热图	30
37	主成分分析图	31
38	基本层次聚类图	32
39	dendrograms() 绘制层次聚类图	33
40	plot	33
41	graphics 包里如何添加图片标题	34
42	ggplot2 包里如何添加图片标题	35

43	par() 函数 mfrow 设置多个图片同个画布	35
44	layout() 设置多个图片同个画布	36
45	cowplot	37
46	gridExtra	38
47	barplot() 水平显示条形图	39
48	coord_flip() 水平显示直方图	40
49	theme_grey() 背景	40
50	theme_gray() 背景	41
51	theme_bw() 背景	41
52	theme_linedraw() 背景	41
53	theme_light() 背景	41
54	ggplot2 包里如何更改背景	42
55	theme_classic() 背景	42
56	theme_dark() 背景	42
57	theme_void() 背景	42
58	去掉背景仅显示坐标轴	43

表目录


1	economics 数据	10
2	1940 年 Virginia 的人口死亡率	12
3	mpg 汽车数据	14
4	VADeaths 数据	14
5	diamonds 数据	16
6	mtcars 数据	20
7	islands 数据	21
8	InsectSprays 数据	22
9	wine 数据	31
10	USArrests 数据	32
11	PlantGrowth 数据	34
12	PlantGrowth 数据	37

引言

手册介绍两部分的内容：基本绘图包和 **ggplot2** 包的绘图。基本绘图包和 **ggplot2** 包绘图各有优势，基本绘图方法运行速度快，**ggplot2** 对于用户来说更容易上手。对于这两种方法谁优谁劣的争论一直都没有消退。具体的讨论可以参见：《Comparing ggplot2 and R Base Graphics》和《Why I don't use ggplot2》。

参考手册的内容会持续更新。有问题请给我发邮件。

版权声明

本手册遵循知识共享许可协议：署名 -非商业性使用 -相同方式共享 3.0 中国大陆 (CC BY-NC-SA 3.0 CN) 

您可以自由地：

- 分享 — 在任何媒介以任何形式复制、发行本作品
- 演绎 — 修改、转换或以本作品为基础进行创作

只要你遵守许可协议条款，许可人就无法收回你的这些权利。

惟须遵守下列条件：

- 署名 — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- 非商业性使用 — 您不得将本作品用于商业目的。
- 相同方式共享 — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

没有附加限制 — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

参考手册使用方法

在后文中会根据不同的绘图任务，分小节介绍一些基本命令。所使用的数据均为 R 中 `datasets` 包的测试数据集。

小标题中的 `graphics::XXXXXX` 代表这个 `XXXXXX` 命令来自 `graphics` 包；`ggplot2::XXXXXX` 代表这个 `XXXXXX` 命令来自于 `ggplot2` 包。（对于一个包 `pkg`，`pkg::name` 会返回在这个 `pkg` 命名空间中的变量名，`pkg:::name` 会返回这个 `pkg` 命名空间中的内部变量名。）

基本绘图命令和 `ggplot2` 包简介

基本绘图命令包 `Graphics` 是默认安装 R 软件时的必须安装的包，配合这个包还要用 `grDevices` 来支持多中颜色和字体的调整。`ggplot2` 是由 Hadley

Wickham开发的基于图层概念的 R 可视化包，一个图层至少包含几何类型、统计类型和位置调整三方面的东西¹。

¹ 该部分内容来源于网络，文档名《ggplot2 作图详解》

其中：几何类型就是指 `geom_` 函数里的东西：

```
## [1] "geom_abline"      "geom_area"
## [3] "geom_bar"         "geom_bin2d"
## [5] "geom_blank"       "geom_boxplot"
## [7] "geom_contour"     "geom_count"
## [9] "geom_crossbar"    "geom_curve"
## [11] "geom_density"     "geom_density_2d"
## [13] "geom_density2d"   "geom_dotplot"
## [15] "geom_errorbar"    "geom_errorbarh"
## [17] "geom_freqpoly"    "geom_hex"
## [19] "geom_histogram"   "geom_hline"
## [21] "geom_jitter"      "geom_label"
## [23] "geom_line"         "geom_linerange"
## [25] "geom_map"          "geom_path"
## [27] "geom_point"        "geom_pointrange"
## [29] "geom_polygon"     "geom_qq"
## [31] "geom_quantile"    "geom_raster"
## [33] "geom_rect"         "geom_ribbon"
## [35] "geom_rug"          "geom_segment"
## [37] "geom_smooth"       "geom_spoke"
## [39] "geom_step"         "geom_text"
## [41] "geom_tile"         "geom_violin"
## [43] "geom_vline"
```

统计类型就是指 `stat_` 函数里的东西：

```
## [1] "stat_bin"          "stat_bin_2d"
## [3] "stat_bin_hex"      "stat_bin2d"
## [5] "stat_binhex"       "stat_boxplot"
## [7] "stat_contour"      "stat_count"
## [9] "stat_density"      "stat_density_2d"
## [11] "stat_density2d"    "stat_ecdf"
## [13] "stat_ellipse"      "stat_function"
## [15] "stat_identity"     "stat_qq"
## [17] "stat_quantile"     "stat_smooth"
## [19] "stat_spoke"         "stat_sum"
## [21] "stat_summary"      "stat_summary_2d"
## [23] "stat_summary_bin"  "stat_summary_hex"
## [25] "stat_summary2d"    "stat_unique"
## [27] "stat_ydensity"
```

图层的位置调整 (`position`) 就是直图形的一些变换方式，例如：

- `dodge` : 躲闪
- `fill` : 填充 (多用于热图)
- `identity` : 不调整位置
- `jitter` : 抖动
- `stack` : 堆栈

散点图

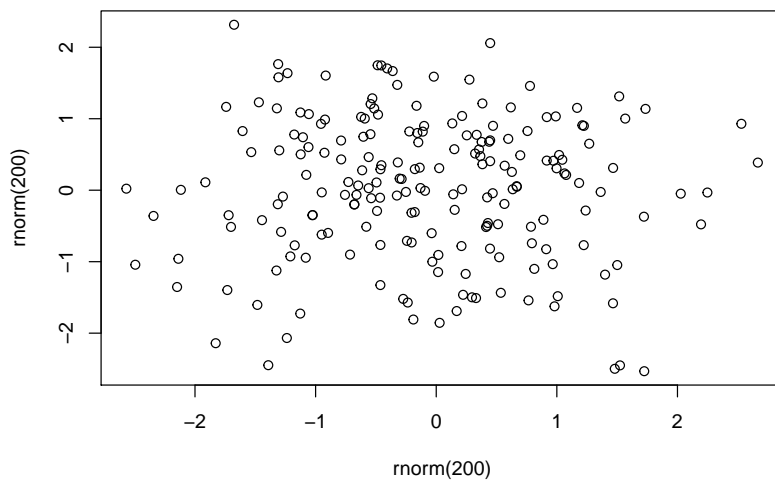
散点图用于研究两组个变量 (x, y) 在坐标平面上的关系。

`graphics::points()` 基本用法

画散点图可以用 `graphics::plot(x,y,type="p")` 或者 `points()` 来实现。首先来看如何用 `plot()` 来画散点图, 在 `plot()` 函数中要加入参数 `type="p"`, 这里的 `p` 就是 `point` 的意思。

```
plot(rnorm(200), rnorm(200), type="p")
```

图 1: 基本散点图; 200 个正态分布的随机数



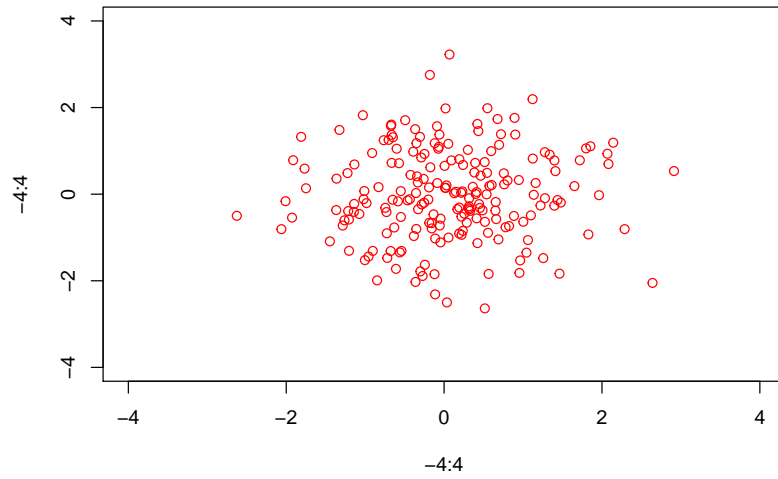
要注意用 `points()` 函数之前需要先生成画板平面。因而建议用 `plot(x,y,type="p")` 方法来画图, 可以少写一行代码。

```
plot(-4:4, -4:4, type = "n") # setting up coord. system
points(rnorm(200), rnorm(200), col = "red")
```

`ggplot2::geom_point()` 基本用法

在 `ggplot2` 里采用 `geom_point()` 函数绘制散点图。

图 2: point 散点图; 200 个正态分布的随机数



```
df <- data.frame(x=rnorm(200), y=rnorm(200))  
ggplot(df, aes(x, y))+geom_point()
```

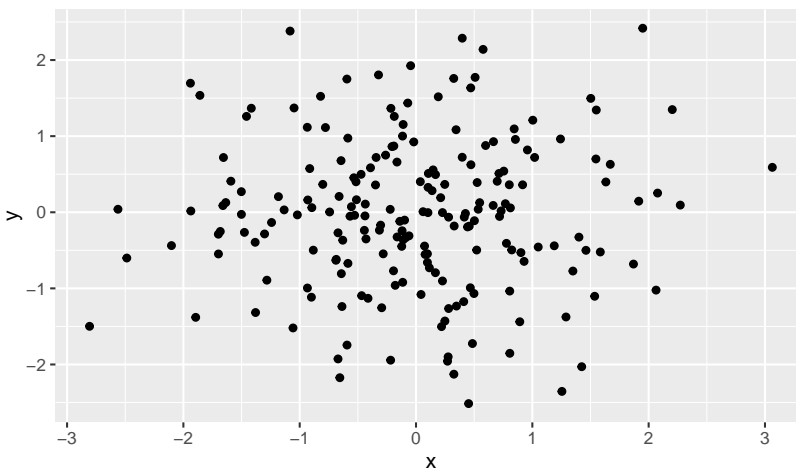


图 3: geom_point 散点图; 200 个正态分布的随机数

折线图

折线图用于显示随某个变量变化的数据。

`graphics::lines()` 基本用法

画折线图同画散点图类似图可以用 `graphics::plot(x,y,type="l")` 或者 `lines()` 来实现。首先来看如何用 `plot()` 来画散点图，在 `plot()` 函数中加入参数 `type="l"`，这里的 `l` 就是 `lines` 的意思。

```
plot(1:10, rnorm(10), type="l")
```

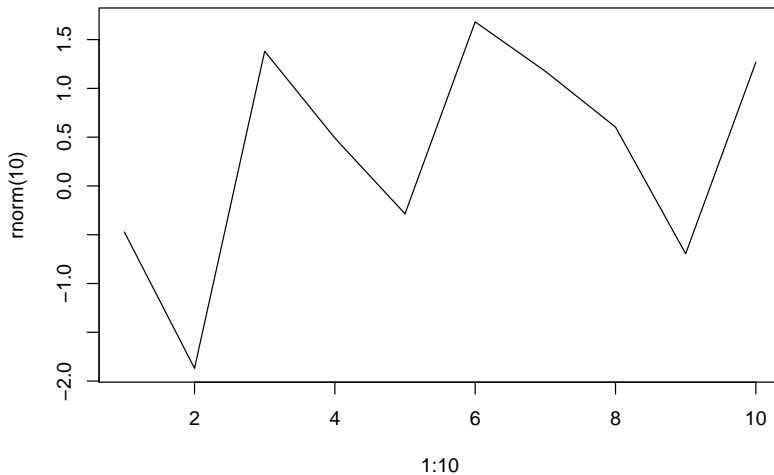


图 4: 基本折线图；10 个正态分布的随机数

在这里我们使用 `cars` 数据集，该数据集记录了刹车时的速度和滑行距离。我们想知道速度同滑行距离之间的关系，用 `lines` 画散点图的平滑线。

```
plot(cars, main = "Stopping Distance versus Speed")
lines(stats::lowess(cars))
```

`ggplot2::geom_line()` 基本用法

在 `ggplot2` 里用 `geom_line()` 函数来连接观测值，这也就是画折线图的方法。

在这里我们使用美国经济数据来展示绘制折线图的方法。`economics` 数据集包含日期，个人消费支出 (`pce`)，人口 (`pop`)，个人存款率 (`pasavert`)，失业人口 (`unemploy`) 和每周失业人口中位数 (`uempmed`)。

表 1: `economics` 数据

date	pce	pop	pasavert	uempmed	unemploy
1967-07-01	507.4	198712	12.5	4.5	2944

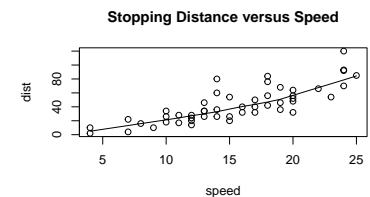


图 5: `lines()` 折线图；刹车速度与滑行距离的关系

date	pce	pop	psavert	uempmed	unemploy
1967-08-01	510.5	198911	12.5	4.7	2945
1967-09-01	516.3	199113	11.7	4.6	2958
1967-10-01	512.9	199311	12.5	4.9	3143
1967-11-01	518.1	199498	12.5	4.7	3066
1967-12-01	525.8	199657	12.1	4.8	3018

在这里我们想知道美国随着事件变化的失业人口情况，用 `geom_line()` 画折线图展示。

```
ggplot(economics, aes(date, unemploy)) + geom_line()
```

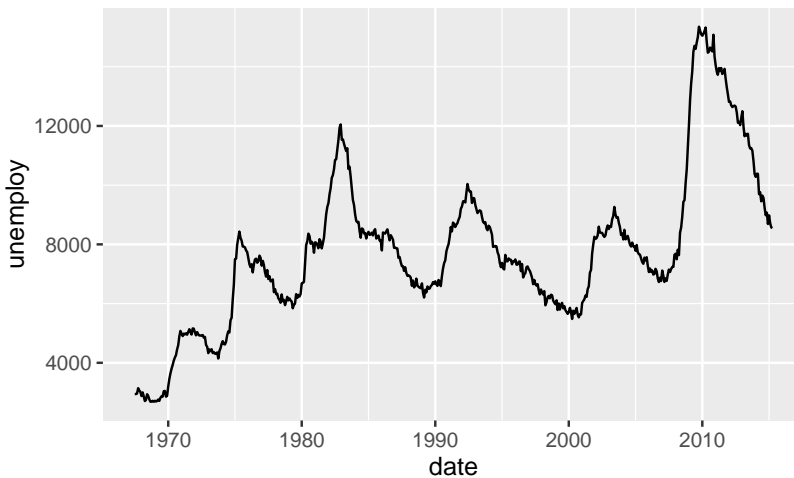


图 6: `geom_line()` 连接观测值；美国人口失业情况折线图

条形图

条形图主要描述一组样本之间某个变量的差异情况。

`graphics::barplot()` 基本用法

下面的代码中会生成 100 个服从泊松分布 $\lambda = 5$ 的随机数，并对随机数做列联表统计，条形图展示了列联表统计的结果。

```
tN <- table(Ni <- stats::rpois(100, lambda = 5))
barplot(tN, col = rainbow(20))
```

`barplot()` 堆栈式条形图

接下来，用 `VADeaths` 数据来绘制一张较为复杂的条形图。`VADeaths` 数据展示了 1940 年 Virginia 的人口死亡率 (Death rates per 1000) 情况，数据根据年龄和人口信息进行的交叉分类。

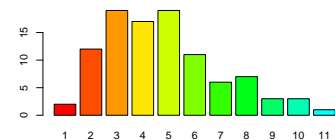


图 7: `barplot()` 基本条形图；统计泊松分布随机数

表 2: 1940 年 Virginia 的人口死亡率

	Rural Male	Rural Female	Urban Male	Urban Female
50-54	11.7	8.7	15.4	8.4
55-59	18.1	11.7	24.3	13.6
60-64	26.9	20.3	37.0	19.3
65-69	41.0	30.9	54.6	35.1
70-74	66.0	54.3	71.1	50.0

对于这个数据，可时化的直接方法是做堆栈式条形图，横坐标是人口信息，纵坐标是根据年龄分成不同颜色的人口数量。

```
barplot(VADeaths)
```



图 8: barplot() 堆栈式条形图；分年龄的人口信息被叠加在一起

`graphics::barplot()` 依次排列的条形图

当然还有一种方法是做按分类排列的条形图 (ggplot2 中的躲闪式 `dodge`)。按分类依次排列的条形图比堆栈式条形图可以更好的比较同个人口信息里的不同年龄层的人口死亡率。

```
barplot(VADeaths, beside = TRUE,
        col = c("lightblue", "mistyrose", "lightcyan",
               "lavender", "cornsilk"),
        legend = rownames(VADeaths), ylim = c(0, 110))
title(main = "Death Rates in Virginia", font.main = 4)
```

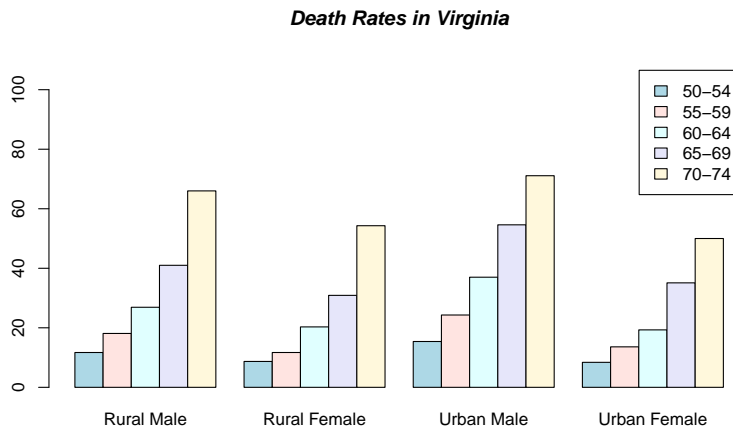


图 9: barplot() 按分类依次排列的条形图

ggplot2::geom_bar() 基本用法

首先看一个在 ggplot2 里的 barplot() 例子：

```
library(ggplot2)
ggplot(mpg, aes(class)) + geom_bar()
```

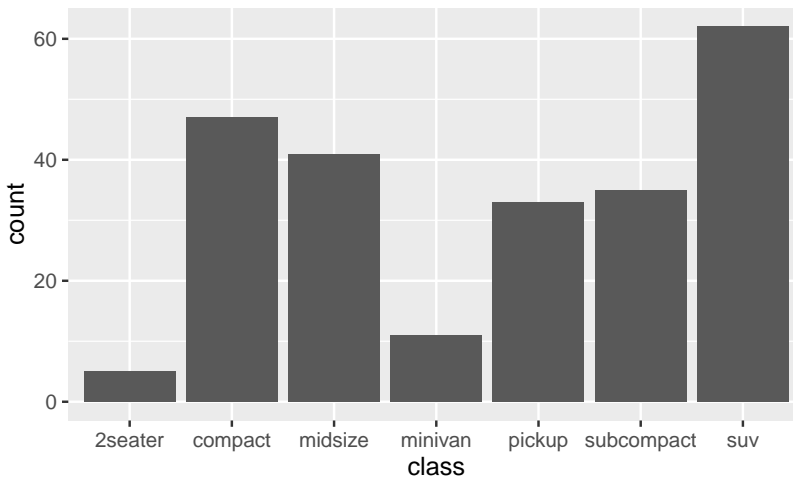


图 10: geom_bar() 基本条形图

要注意这个数据表的格式：每一行为一个观测（样本），每一列是一个变量（特征）。这是 ggplot2 默认的输出格式。

表 3: mpg 汽车数据

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
10	audi	a4 quattro	2.0	2008	4	manual(m6)	4	20	28	p	compact
11	audi	a4 quattro	2.0	2008	4	auto(s6)	4	19	27	p	compact
12	audi	a4 quattro	2.8	1999	6	auto(l5)	4	15	25	p	compact
13	audi	a4 quattro	2.8	1999	6	manual(m5)	4	17	25	p	compact
14	audi	a4 quattro	3.1	2008	6	auto(s6)	4	17	25	p	compact
15	audi	a4 quattro	3.1	2008	6	manual(m6)	4	15	25	p	compact
16	audi	a6 quattro	2.8	1999	6	auto(l5)	4	15	24	p	midsize
17	audi	a6 quattro	3.1	2008	6	auto(s6)	4	17	25	p	midsize
18	audi	a6 quattro	4.2	2008	8	auto(s6)	4	16	23	p	midsize
19	chevrolet	c1500 suburban 2wd	5.3	2008	8	auto(l4)	r	14	20	r	suv
20	chevrolet	c1500 suburban 2wd	5.3	2008	8	auto(l4)	r	11	15	e	suv

数据格式转换

`barplot()` 小节中的 `VADeaths` 数据集，每一行是年龄层，每一列是不同地区性别的人口，这样的数据是不符合 `ggplot2` 的输入条件的，如果想用 `ggplot2` 绘图就必须先做转换。做数据格式转换所用的命令是 `reshape2` 包里的 `melt`，这个包也是 Hadley Wickham 开发的。

表 4: VADeaths 数据

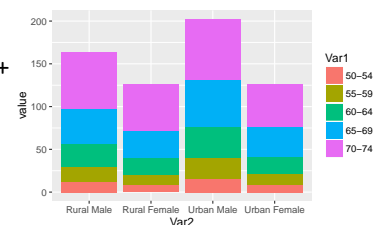
Var1	Var2	value
50-54	Rural Male	11.7
55-59	Rural Male	18.1
60-64	Rural Male	26.9
65-69	Rural Male	41.0
70-74	Rural Male	66.0
50-54	Rural Female	8.7

可以看到转换后的表格，一共有三个变量，第一列是年龄层信息 (`Var1`)，第二列是地区性别信息 (`Var2`)，第三列是死亡率信息 (`value`)。每一行是一个变量。

`ggplot2::geom_bar()` 堆栈式条形图

有了这样的数据表，我们可以画出 `barplot()` 小节中的堆栈式条形图和依次排列的条形图。

```
ggplot(data=VADeaths.ggplot2, aes(x=Var2, y=value, fill=Var1)) +
  geom_bar(stat="identity")
```

图 11: `geom_bar()` 堆栈式条形图

`ggplot2::geom_bar()` 依次排列的条形图

调节条形图的样式所用的命令就是在 `position` 里变换不同的位置状态。条形图默认的样式是堆栈式。

```
ggplot(data=VADeaths.ggplot2, aes(x=Var2, y=value, fill=Var1)) +  
  geom_bar(stat="identity", position="dodge")
```

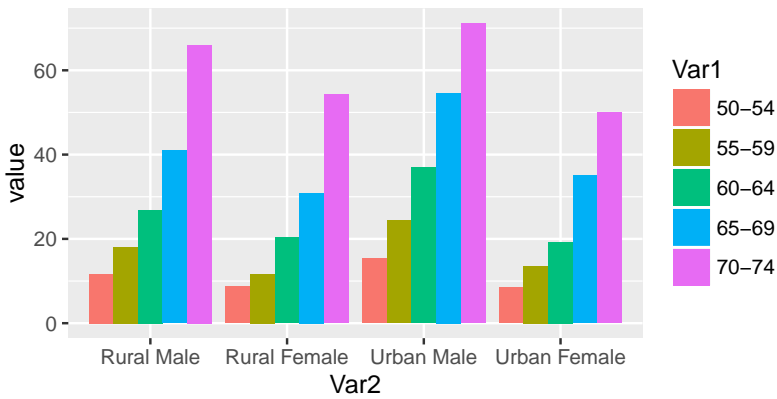


图 12: `geom_bar()` 依次排列式条形图

`ggplot2::geom_bar()` 比例式条形图

比例式条形图统计的是在某个变量下，所有可能情况所占的比例。将 `position` 设置为 `fill` 模式即可。

```
ggplot(data=VADeaths.ggplot2, aes(x=Var2, y=value, fill=Var1)) +  
  geom_bar(stat="identity", position="fill")
```

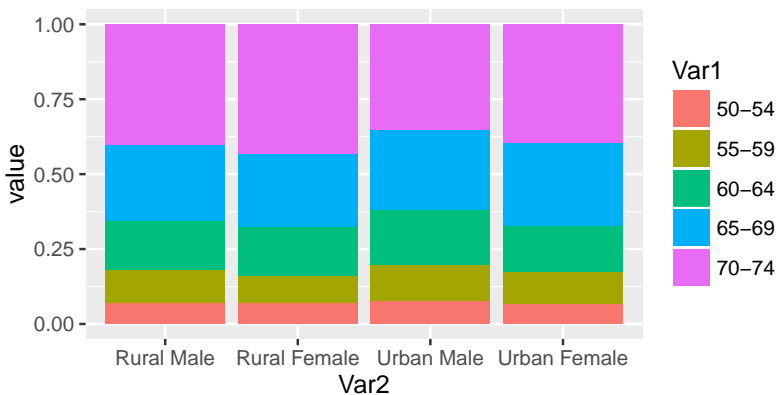


图 13: `geom_bar()` 比例式条形图

面积图

面积图表示一个连续变量的变化程度，同时也展示了部分与整体之间的关系。

`graphics::polygon()` 基本用法

在基本绘图包中画面积相关的图形等同于画多边形。

这次我们用个钻石相关的数据来做展示，这个数据集包含了 54000 个钻石的价格以及其他相关指标。

表 5: diamonds 数据

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
0.24	Very Good	J	VVS2	62.8	57	336	3.94	3.96	2.48
0.24	Very Good	I	VVS1	62.3	57	336	3.95	3.98	2.47
0.26	Very Good	H	SI1	61.9	55	337	4.07	4.11	2.53
0.22	Fair	E	VS2	65.1	61	337	3.87	3.78	2.49
0.23	Very Good	H	VS1	59.4	61	338	4.00	4.05	2.39

我们统计随着价格 (price) 的变化，钻石的的评价 (cut: quality of the cut) 的趋势²

可以看到如果用 `density()` 外加 `polygon()` 图形并不好看。后一个密度图会覆盖前一个图像。

```
d <- density(diamonds[diamonds$cut=="Ideal",]$price)
plot(d,main="",xlab = "Price")
polygon(d, col="red",border = "red")
d <- density(diamonds[diamonds$cut=="Premium",]$price)
polygon(d, col="yellow",border = "yellow")
d <- density(diamonds[diamonds$cut=="Good",]$price)
polygon(d, col="green",border = "green")
d <- density(diamonds[diamonds$cut=="Very Good",]$price)
polygon(d, col="black",border = "black")
d <- density(diamonds[diamonds$cut=="Fair",]$price)
polygon(d, col="orange",border = "orange")
```

画堆积面积图要求每一个面积图堆积到前一个面积图的上方，在 `graphics` 包中没有专门的函数对应。

下面所用的方法参考自 R 邮件列表中的一封信³。

代码如下所示：

² 想画更漂亮的面积堆积图请参考这个介绍《Data mountains and streams - stacked area plots in R》。

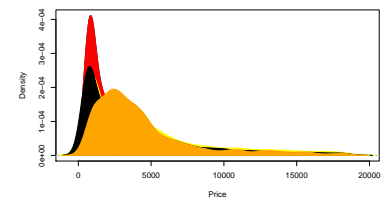


图 14: `polygon()` 密度图

³ R-help 邮件列表 Stacked Area chart


```

stackedPlot <- function(data, time=NULL, col=1:length(data), ...){
  if (is.null(time))
    time <- 1:length(data[[1]]);
  plot(0, 0, xlim = range(time), ylim = c(0,max(rowSums(data))), t="n", ...);
  for (i in length(data):1) {
    # Die Summe bis zu aktuellen Spalte
    prep.data <- rowSums(data[1:i]);
    # Das Polygon muss seinen ersten und letzten Punkt auf der Nulllinie haben
    prep.y <- c(0, prep.data, 0)
    prep.x <- c(time[1], time, time[length(time)])
    polygon(prep.x, prep.y, col=col[i], border = NA);
  }
}
diamonds.data <- as.data.frame.matrix(t(table(diamonds$cut,diamonds$price)))
stackedPlot(diamonds.data)

```

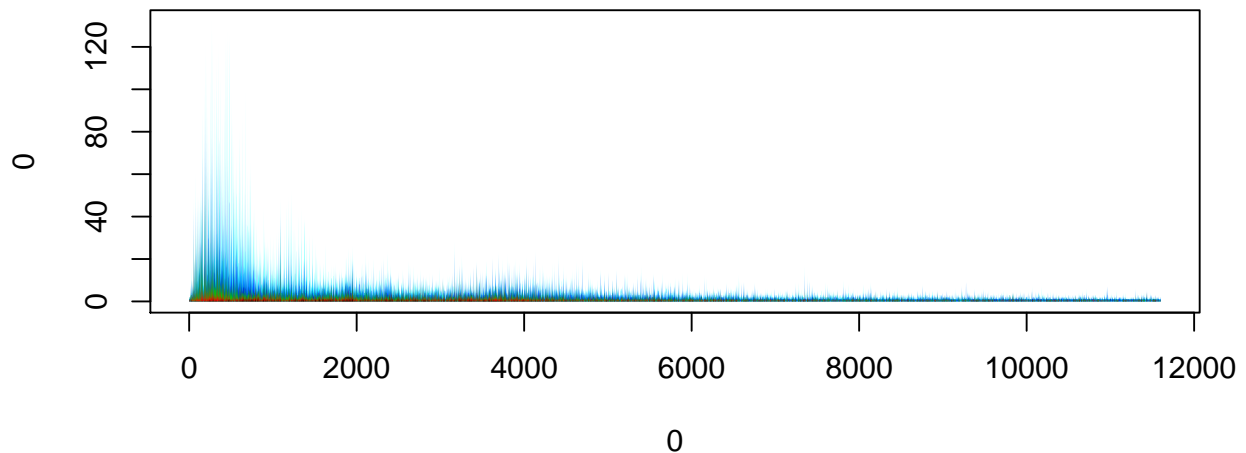


图 15: polygon() 面积堆积图

ggplot2::geom_area() 基本用法

在 ggplot2 里画面积图就非常简单了，采用 geom_area() 函数就可以生成堆积面积图。

```

ggplot(diamonds, aes(x = price, fill = cut))+
  geom_area(stat = "bin")

```

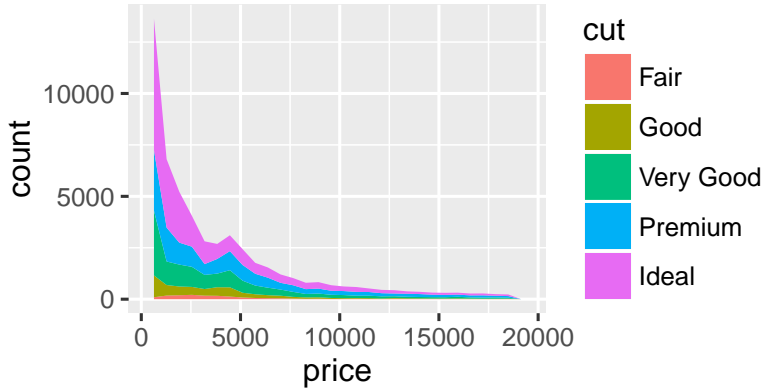


图 16: geom_area() 堆积面积图

密度估计图

用 `graphics` 画密度估计图

其实在上一个小节已经提到了绘制密度图像。用 `graphics` 包画密度图像就是直接 `plot(density(data))`

例如我们取 200 个正态分布的随机数，画其密度估计图像。

```
set.seed(1234)
rating <- rnorm(200)
plot(density(rating))
```

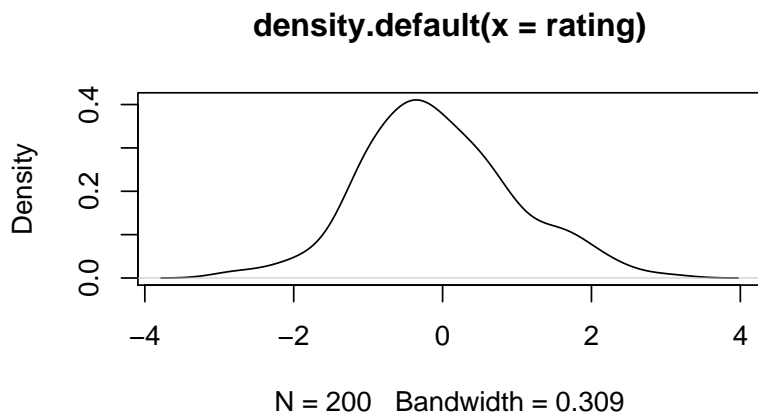


图 17: 密度估计图

如果需要在一张画布上画两个密度图。直接叠加就可以，如果想把密度函数的范围控制的更准确一些，可以参考博客《Plotting two or more overlapping density plots on the same graph》的内容⁴。个人感觉用 `graphics` 包来画密

⁴ <http://onertipaday.blogspot.com/2007/09/plotting-two-or-more-overlapping.html>

度函数图像要比用 `ggplot2` 容易上手。

```
set.seed(1234)
rating <- rnorm(200)
rating2 <- rnorm(200, mean=.8)

plot(density(rating))
lines(density(rating2), col="red")
```

`ggplot2::geom_density()`

在 `ggplot2` 里有专门的密度函数计算和画图函数 `geom_density()`。所以不用特意用 `density()` 函数来计算密度函数。在这里用 `diamonds` 数据做例子，我们想知道 `depth` 的分布情况，并且根据不同的 `cut` 信息分别画出对应的密度函数图。

```
ggplot(diamonds, aes(depth, colour = cut)) +
  geom_density()
```

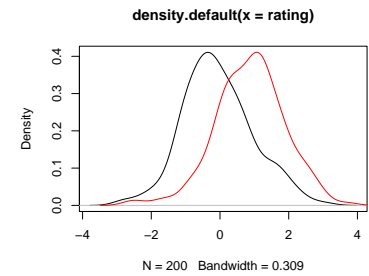
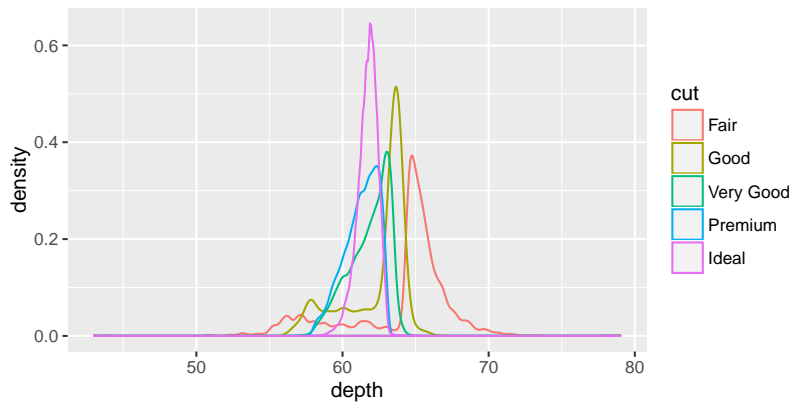


图 18: 两个核密度估计图

图 19: `geom_density()` 核密度估计图

频率图像

频率图像同密度函数图像的区别是：前者统计出现的频数，后者统计概率密度函数。从图中直观的反应就是纵坐标的单位不一样。

用 `graphics` 包画频率图像

用 `graphics` 包画频率图像没有专门的函数，需要组合一下现有的工具。在这里，我们用 `mtcars` 的数据来实现，`mtcats` 数据是 1974 年 Motor Trend 杂志所刊登的一组 32 不同种类的汽车耗油量和特征信息。

表 6: mtcars 数据

	mpg	cyl	disp	hp	drat
Mazda RX4	21.0	6	160	110	3.90
Mazda RX4 Wag	21.0	6	160	110	3.90
Datsun 710	22.8	4	108	93	3.85
Hornet 4 Drive	21.4	6	258	110	3.08
Hornet Sportabout	18.7	8	360	175	3.15

我们想知道耗油量（每英里加仑数 mpg）的频数分布图。所首先要用 `hist()` 函数来获得密度函数信息，计算出频数同密度值的比例。

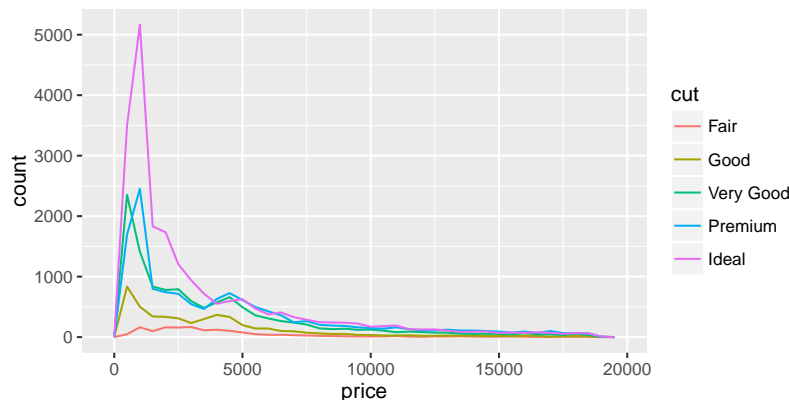
```
myhist <- hist(mtcars$mpg, plot = FALSE)
multiplier <- myhist$counts / myhist$density
mydensity <- density(mtcars$mpg)
mydensity$y <- mydensity$y * multiplier[1]
```

```
plot(mydensity)
```

用 `ggplot2::geom_freqpoly()` 画频率图像

在这里用 `diamonds` 数据做例子，我们想知道不同 `price` 的出现频数情况，并且根据不同的 `cut` 信息分别画出对应的密度函数图。

```
ggplot(diamonds, aes(price, colour = cut)) +
  geom_freqpoly(binwidth = 500)
```



直方图

直方图是一种对数据分布情况的图形表示，它的样子同条形图相似，但直方图是用面积而非单一的高度来表示数量（同分布相关的图，都是用面积来表示数量）。

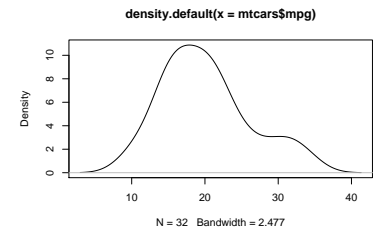


图 20: 用 Graphics 函数画频率图

图 21: `geom_freqpoly()` 频率图

`graphics::hist()` 基本用法

我们用世界主要大陆地区的数据来做演示，`islands` 数据统计了主要大陆和岛屿的面积信息。

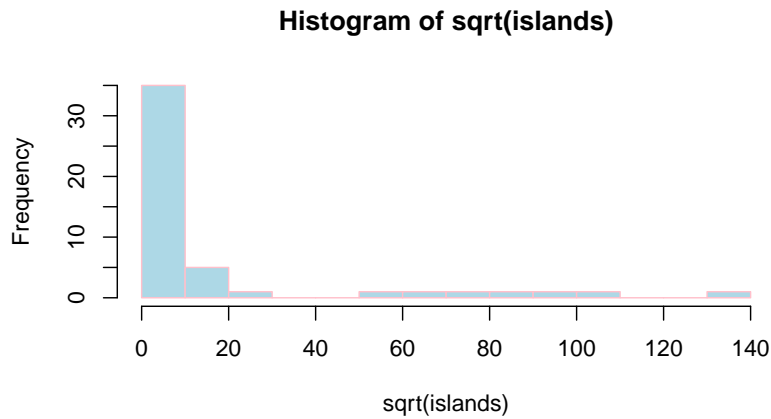
表 7: `islands` 数据

islands[1:10]	
Africa	11506
Antarctica	5500
Asia	16988
Australia	2968
Axel Heiberg	16
Baffin	184
Banks	23
Borneo	280
Britain	84
Celebes	73

绘图命令为：

```
hist(sqrt(islands), breaks = 12, col = "lightblue", border = "pink")
```

图 22: `hist()` 直方图



`ggplot2::geom_hist()` 基本用法

用 `ggplot2` 画直方图时首先要确保输入的数据是数据框 (`data.frame`) 格式，在这里我们需要先把 `islands` (`vector`) 转换成数据框。

```
ggplot(as.data.frame(islands), aes(sqrt(islands))) +
  geom_histogram()
```

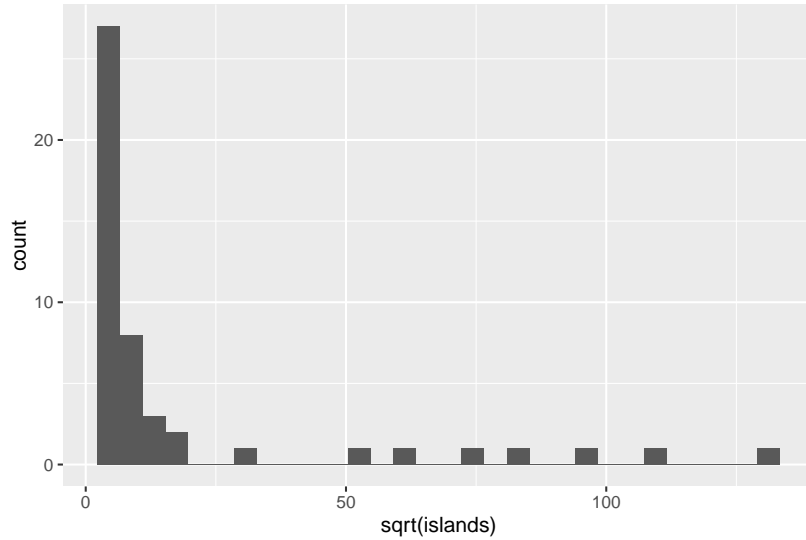


图 23: geom_hist() 直方图

箱线图

箱线图是利用数据中的五个统计量（从下往上依次是）：最小值、第一四分位数、中位数、第三四分位数与最大值来描述数据的一种方法，它也可以粗略地看出数据是否具有有对称性，分布的分散程度等信息，特别可以用于对几个样本的比较。

`graphics::boxplot()` 基本用法

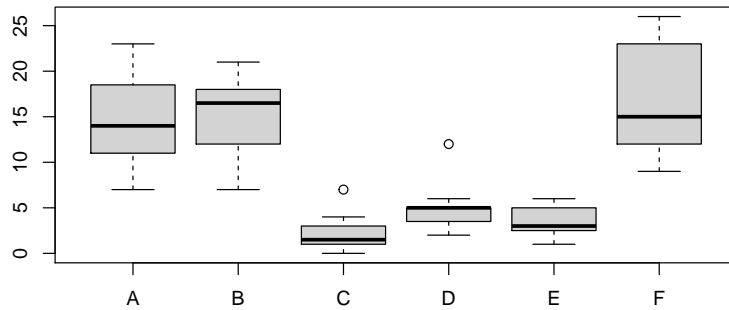
我们采用的数据是 `InsectSprays`（昆虫喷雾剂的有效性）列表，统计的内容为不同杀虫剂处理的农业试验单位的昆虫数量。表的第一列是昆虫数量，第二列是喷雾器种类。

表 8: `InsectSprays` 数据

count	spray
10	A
7	A
20	A
14	A
14	A
12	A

从数据中我们想知道不同的杀虫剂可以处理掉的每实验单位的昆虫有多少。这就用到了 `boxplot()`，来画箱线图。用 `graphics::boxplot()` 包来画箱线图要比 `ggplot2` 简单很多，并且图里带有 `error bar`。

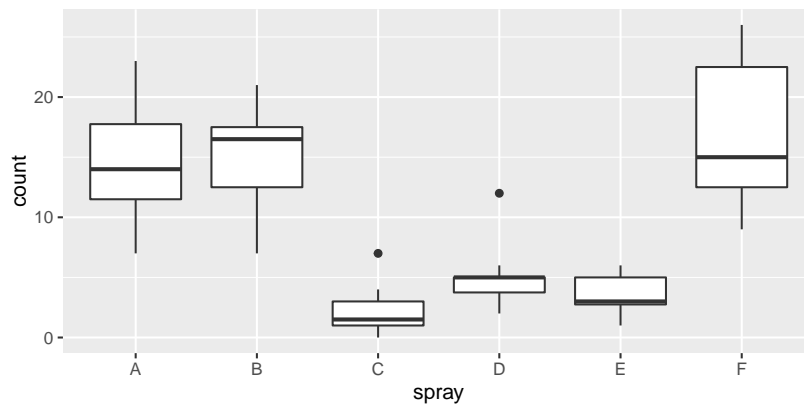
```
boxplot(count ~ spray, data = InsectSprays, col = "lightgray")
```

图 24: `boxplot()` 箱线图

`graphics::geom_boxplot()` 基本用法

从图中可以观察到 `geom_boxplot()` 默认的箱线图是不带 `error bar` 的。如果想在图中画上 `error bar`，需要使用 `stat_boxplot()` 函数。

```
ggplot(InsectSprays, aes(spray, count))+geom_boxplot()
```

图 25: `geom_boxplot()` 箱线图

用 `graphics::stat_boxplot()` 为箱线图添加 `error bar`

在添加 `error bar` 时要注意代码的顺序，要记住 `ggplot2` 里的绘图都是一层一层叠加上的，也就是加号后面的函数绘制的图层会叠加到加号之前的图层上面。如果 `geom_boxplot()` 在 `stat_boxplot()` 函数之前，那么程序运行的时候，会先画箱线图，后画 `error bar`。这时，我们会看见一条 `error bar` 的垂直线贯穿箱线图，并不美观。

```
ggplot(InsectSprays, aes(spray, count))+
  geom_boxplot()+
  stat_boxplot(geom = 'errorbar', width=0.5)
```

所以正确的绘图方法时先画 error bar，再画箱线图。(注意下面代码的顺序)

```
ggplot(InsectSprays, aes(spray, count))+
  stat_boxplot(geom = 'errorbar', width=0.5)+
  geom_boxplot()
```

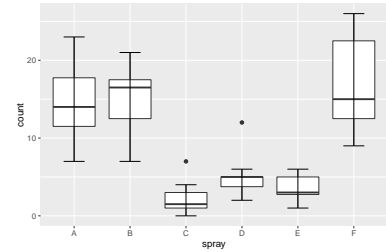
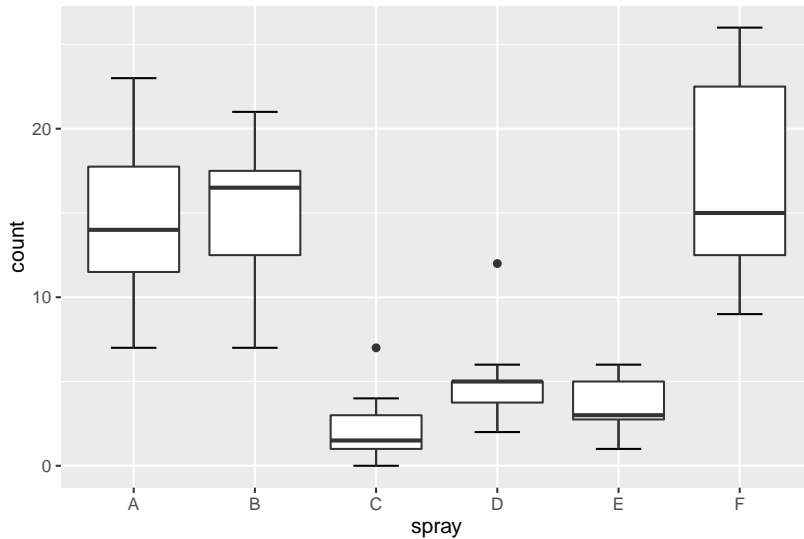


图 26: 错误的 error bar 箱线图

图 27: 带 error bar 的箱线图

提琴图

提琴图展示了数据的密度估计情况，同箱线图类似。但是箱线图只是展示了分位数的位置，而提琴图展示了任意位置的数据密度。

`vioplot::vioplot()` 基本用法

在基本绘图包中没有绘制提琴图的方法，在这里需要安装 `vioplot` 包来绘制提琴图。

在这里我们使用 `mtcars` 的数据来展示在不同的发动机汽缸数量 (`cyl`) 下每英里耗油量 (`mpg`) 的分布情况。

```
library(vioplot)
x1 <- mtcars$mpg[mtcars$cyl==4]
x2 <- mtcars$mpg[mtcars$cyl==6]
x3 <- mtcars$mpg[mtcars$cyl==8]
vioplot(x1, x2, x3, names=c("4 cyl", "6 cyl", "8 cyl"),
```



```
col="gold")
title("Violin Plots of Miles Per Gallon")
```

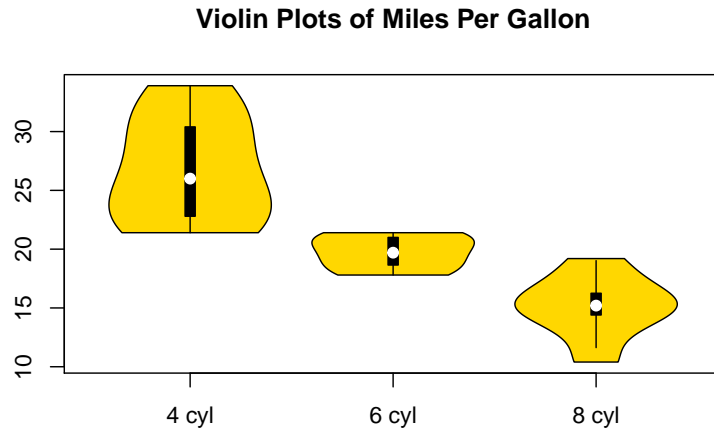


图 28: vioplot() 提琴图

ggplot2::geom_violin() 基本用法

用 *ggplot2* 来画提琴图比较容易, 不需要向上面一样将数据提取数来, 只要在绘制时定义好美学映射 *aes()*, 通俗来讲, 即图像的横轴和纵轴所代表的的数据。

```
ggplot(mtcars, aes(factor(cyl), mpg))+ geom_violin()
```

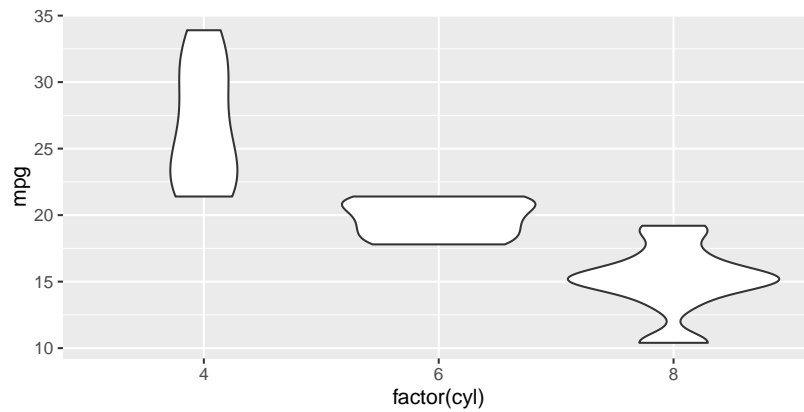


图 29: geom_violin() 提琴图

在提琴图中添加箱线图

上面的提琴图同 `vioplot` 包里的提琴图还有些区别。如果我们在提琴图里展示箱线图的信息，只需要在代码的最后添加一个箱线图即可。

```
ggplot(mtcars, aes(factor(cyl), mpg))+
  geom_violin()+
  geom_boxplot(width=.1)
```

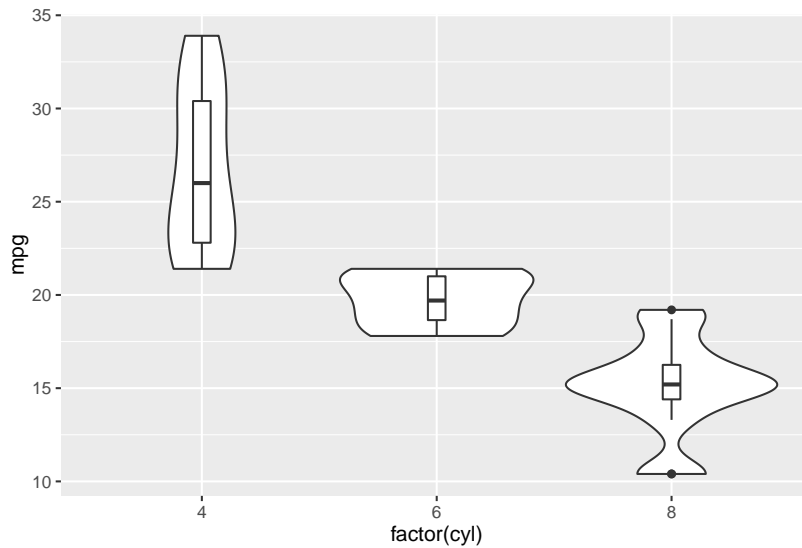


图 30: 添加箱线图信息的提琴图

在提琴图中添加均值和标准差信息

这时需要用到 `ggplot2::stat_summary()` 函数，在函数中我们定义需要计算的信息时均值和一倍标准差 `fun.data=mean_sdl, fun.args = list(mult = 1)`。

```
ggplot(mtcars, aes(factor(cyl), mpg))+
  geom_violin()+
  stat_summary(fun.data = mean_sdl,
              geom = "pointrange",
              color = "red",
              fun.args = list(mult = 1))
```

```
## Warning: Computation failed in `stat_summary()`:
## Hmisc package required for this function
```

Cleveland 点图

Cleveland 点图用于绘制有分类别的数据信息。为什么用点来表示呢？因为 Cleveland 认为点图清晰容易绘制。

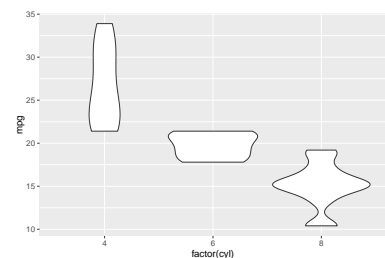
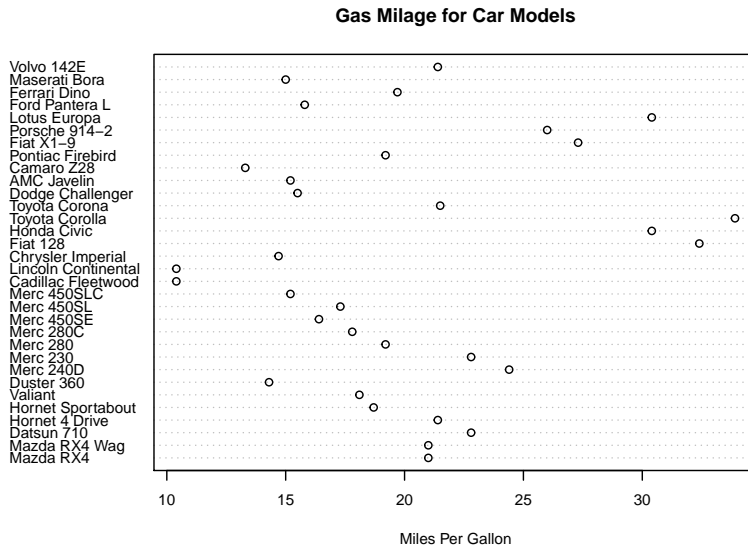


图 31: 添加均值和标准差信息的提琴图

`graphics::dotchart()`

点图通过点的位置信息展示数据。下图中我们使用 `mtcars` 数据来绘制关于每英里耗油量的点图。

```
dotchart(mtcars$mpg, labels=row.names(mtcars), cex=.7,
         main="Gas Milage for Car Models",
         xlab="Miles Per Gallon")
```

图 32: `dotchart()` 绘制 Cleveland 点图`ggplot2::geom_dotplot()`

下图中采用 `geom_dotplot()` 来绘制点图, 使用了 `fill = row.names(mtcars)` 参数来填充颜色, 其实在这幅图里颜色信息同 `y` 坐标信息是冗余的, 所以最好不用颜色。在 `ggplot2` 中添加数据分组颜色的一种方法就是用 `fill` 参数, 这个参数在其他图表中也适用。

```
ggplot(mtcars, aes(x = mpg, y = row.names(mtcars), fill = row.names(mtcars))) +
  geom_dotplot(binaxis = "y",
              stackgroups = TRUE,
              binwidth = 1,
              method = "histodot")
```

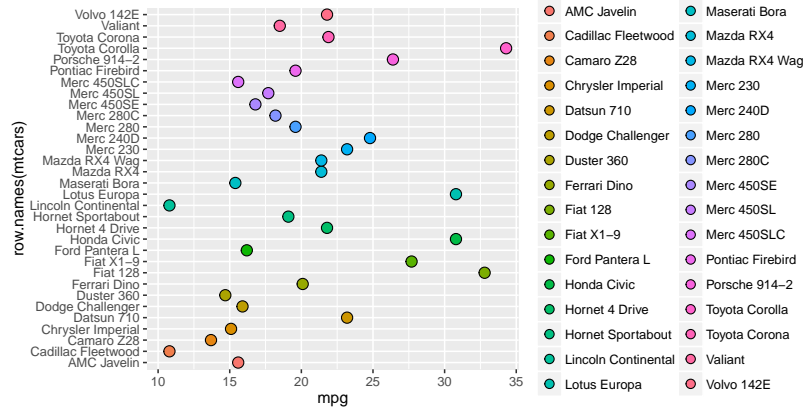


图 33: geom_dotplot() 绘制 Cleveland 点图

热图

graphics::heatmap() 基本用法

heatmap() 是十分简单的绘制热图的方法，也有很多可以调整的参数，在绘制热图时首选使用这个函数。

```
x <- as.matrix(mtcars)
rc <- rainbow(nrow(x), start = 0, end = .3)
cc <- rainbow(ncol(x), start = 0, end = .3)
hv <- heatmap(x,
  col = cm.colors(256),
  scale = "column",
  RowSideColors = rc,
  ColSideColors = cc,
  margins = c(5,10),
  xlab = "specification variables",
  ylab = "Car Models",
  main = "Heatmap of Mtcars data")
```

ggplot2::geom_tile() 基本用法

用 ggplot2 画 heatmap 没有现成的函数，比较漂亮的方法是参考《ggplot2: Quick Heatmap Plotting》在这里仅提供一个例子，不做赘述，另外强烈不建议使用 ggplot2 来画 heatmap。

```
library(reshape2)
library(ggplot2)
dat <- matrix(rnorm(100, 3, 1), ncol=10)
names(dat) <- paste("X", 1:10)
dat2 <- melt(dat, id.var = "X1")
```

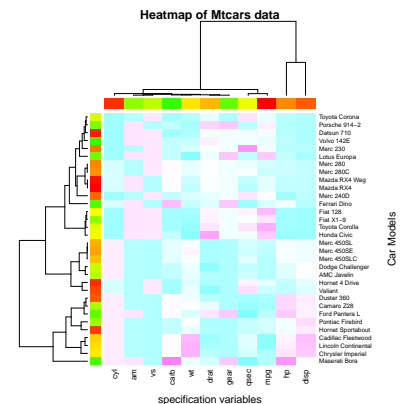


图 34: 用 heatmap() 绘制热图

```
ggplot(dat2, aes(as.factor(Var1), Var2, group=Var2)) +
  geom_tile(aes(fill = value)) +
  geom_text(aes(fill = dat2$value, label = round(dat2$value, 1))) +
  scale_fill_gradient(low = "white", high = "red")
```

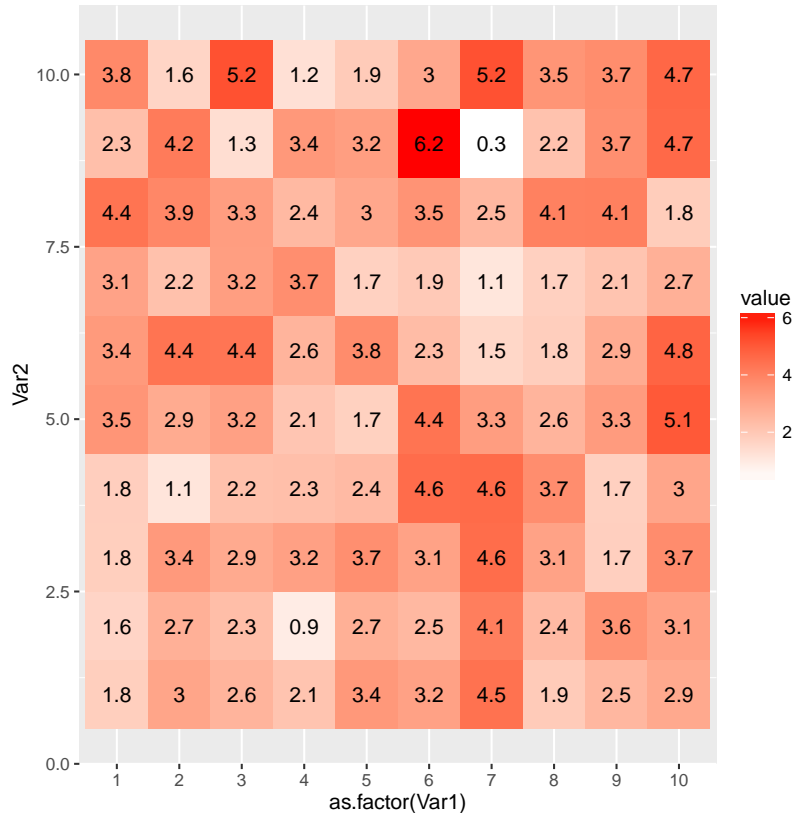


图 35: geom_tile() 绘制热图

pheatmap::pheatmap() 基本用法

最容易使用且功能全面的热图绘制包是 *pheatmap*，它提供了 `cluster_rows=FALSE`，`cluster_cols=TRUE` 两个参数来控制是否对每一行/每一列进行聚类。

```
library(pheatmap)
test = matrix(rnorm(200), 20, 10)
test[1:10, seq(1, 10, 2)] = test[1:10, seq(1, 10, 2)] + 3
test[11:20, seq(2, 10, 2)] = test[11:20, seq(2, 10, 2)] + 2
test[15:20, seq(2, 10, 2)] = test[15:20, seq(2, 10, 2)] + 4
colnames(test) = paste("Test", 1:10, sep = "")
rownames(test) = paste("Gene", 1:20, sep = "")
# 设置每一列的注释
annotation_col = data.frame(
```

```

CellType = factor(rep(c("CT1", "CT2"), 5)),
Time = 1:5
)
rownames(annotation_col) = paste("Test", 1:10, sep = "")
# 设置每一行的注释
annotation_row = data.frame(
  GeneClass = factor(rep(c("Path1", "Path2", "Path3"), c(10, 4, 6)))
)
rownames(annotation_row) = paste("Gene", 1:20, sep = "")
# 设置注释的颜色
ann_colors = list(
  Time = c("white", "firebrick"),
  CellType = c(CT1 = "#1B9E77", CT2 = "#D95F02"),
  GeneClass = c(Path1 = "#7570B3", Path2 = "#E7298A", Path3 = "#66A61E")
)
pheatmap(test,
  annotation_col = annotation_col,
  annotation_row = annotation_row,
  annotation_colors = ann_colors)

```

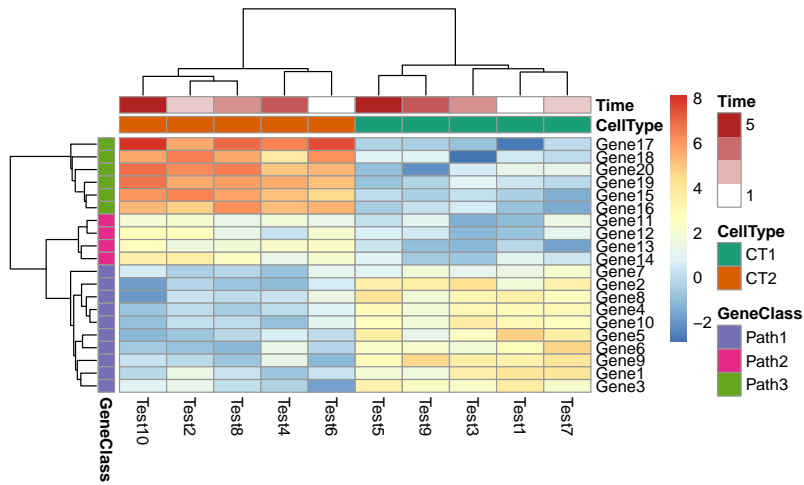


图 36: pheatmap() 绘制热图

PCA 第一第二主成分平面图

做 PCA 时我们通常会前两个主成分展示到坐标平面上，以此来区分样本的差异性。这种图是基本统计图形的综合展示。

`ggbiplot::ggbiplot()` 基本用法

在这里强烈推荐使用 `ggbiplot` 包来画主成分分析的图。该包没有推送到 CRAN，要从 Github 安装它。

```
library(devtools)
install_github("vqv/ggbiplot")
```

我们用 `ggbiplot` 包里的 `wine` 数据来做主成分分析，该数据记录了意大利同一个地区的三种葡萄酒的化学成分和其他特征⁵。

⁵ 详情参见 Wine Data Set

表 9: wine 数据

Alcohol	MalicAcid	Ash	AlcAsh	Mg
14.23	1.71	2.43	15.6	127
13.20	1.78	2.14	11.2	100
13.16	2.36	2.67	18.6	101
14.37	1.95	2.50	16.8	113
13.24	2.59	2.87	21.0	118
14.20	1.76	2.45	15.2	112

用 `ggbiplot` 包画主成分分析图的好处在于，它可以自动在横纵坐标标签的位置填写该主成分所占比例。

```
library(ggbiplot)
wine.pca <- prcomp(wine, scale. = TRUE)
ggbiplot(wine.pca, obs.scale = 1, var.scale = 1,
  groups = wine.class, ellipse = TRUE, circle = TRUE) +
  scale_color_discrete(name = '')
```

层次聚类图

层次聚类是聚类算法的一种，通过计算样本间的相似度来构造一棵聚类树。

`plot.hclust()` 基本用法

这个说的是用 `hclust` 类的 S3 method 来画层次聚类图。例如我们前面画直方图的 `hist()` 函数也是 `histogram` 类的 S3 method。

我们采用美国暴力犯罪率来展示层次聚类，`USArrests` 包含每个州的三种犯罪人员被逮捕的数量以及该州城市地区人口数量。

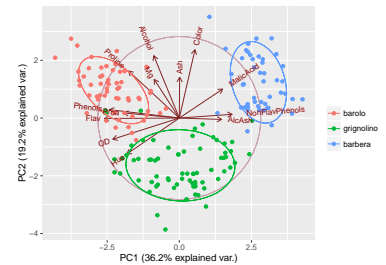


图 37: 主成分分析图

表 10: USArrests 数据

	Murder	Assault	UrbanPop	Rape
Alabama	13.2	236	58	21.2
Alaska	10.0	263	48	44.5
Arizona	8.1	294	80	31.0
Arkansas	8.8	190	50	19.5
California	9.0	276	91	40.6
Colorado	7.9	204	78	38.7

简单的聚类图如下所示

```
hc <- hclust(dist(USArrests), "ave")
plot(hc)
```

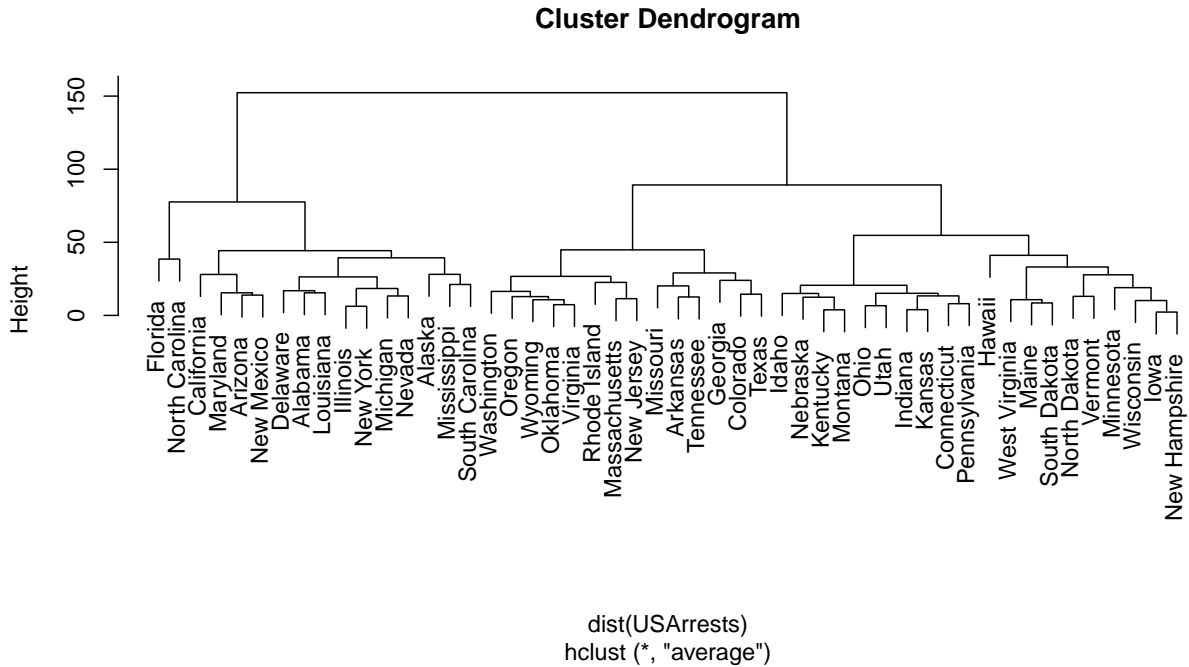


图 38: 基本层次聚类图

用 `ggdendro::dendrograms` 包画层次聚类图

`ggdendro` 包是基于 `ggplot2` 包的基础上来画图。rpubs 网站里有个 Gaston Sanchez 撰写的非常好的使用指南，该指南包括普通的 `plot.hclust()` 和 `dendrograms()` 的使用方法以及系统发育树的画法⁶。关于 `dendrograms()` 的基本用法可以参见它的 vignettes⁷。

下面用 `USArrests` 的数据画一个旋转的层次聚类图。

⁶ Visualizing Dendrograms in R

⁷ Using the `ggdendro` package to plot dendrograms


```
library(ggdendro)
hc <- hclust(dist(USArrests), "ave")
hcdata <- dendro_data(hc)
ggdendrogram(hcdata, rotate=TRUE, size=2) + labs(title="Dendrogram in ggplot2")
```

Dendrogram in ggplot2

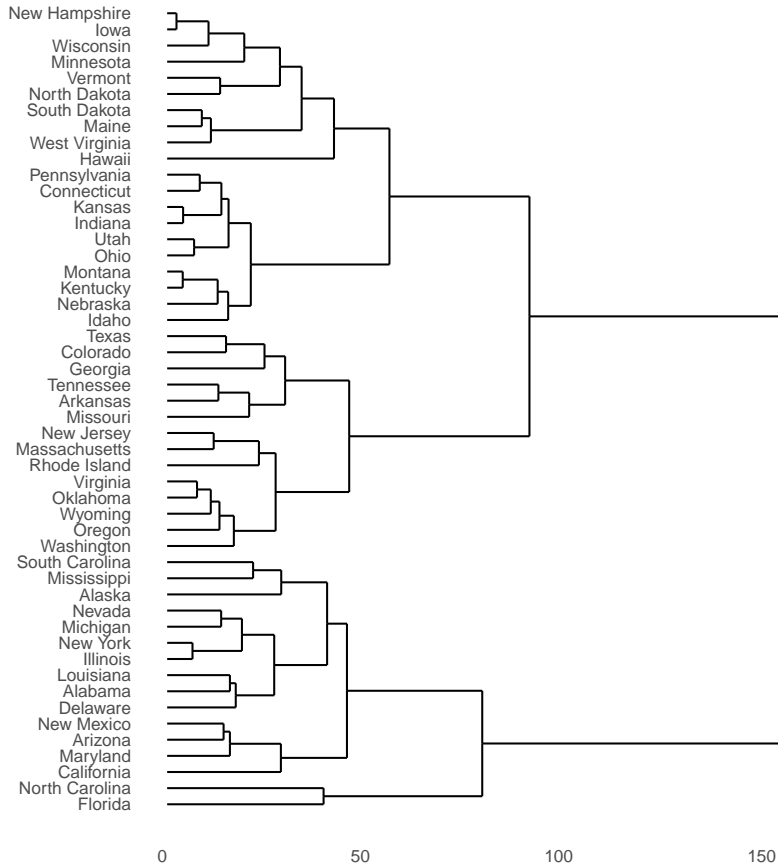


图 39: dendrograms() 绘制层次聚类图

用 *ape* 包绘制系统发育树

ape 包提供了将普通的层次聚类数据转换成系统发育树的工具，然后用 *phylo* 类的 S3 method 来画图。

下图是将 *USArrests* 层次聚类的结果转换成扇形发育树的图。

```
hc <- hclust(dist(USArrests), "ave")
library(ape)
plot(as.phylo(hc), type = "fan")
```

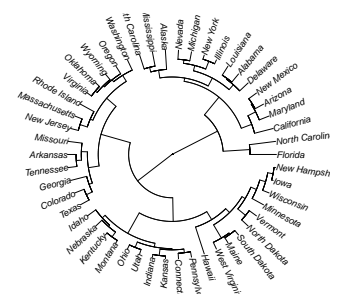


图 40: plot.phylo() 绘制层次聚类图

关于绘图，我们还关心的其他问题

如何添加 *title*，*x* 轴和 *y* 轴标签

在 `graphics` 包中添加标题用 `main` 参数，添加子标题用 `sub` 参数，添加 *x* 轴标签用 `xlab` 参数，添加 *y* 轴标签用 `ylab` 参数。

```
plot(table(rpois(100, 5)), type = "h", col = "red", lwd = 10,
      main = "rpois(100, lambda = 5)", sub="this is a sub title",
      xlab="x axis title", ylab="y axis title"
    )
```

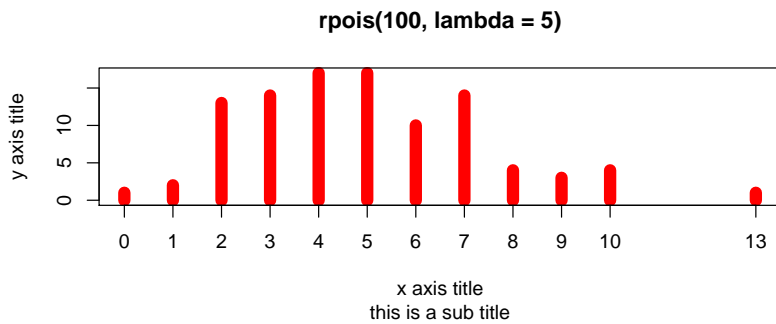


图 41: `graphics` 包里如何添加图片标题

在 `ggplot2` 中可以用 `ggtitle()`、`xlab()` 以及 `ylab()` 这三个函数来添加标题和坐标轴标签。

这里我们又用了一个新的示例数据 `PlantGrowth`，该数据展示了在一个试验中控制不同的条件下植物的生长情况。

表 11: `PlantGrowth` 数据

weight	group
4.17	ctrl
5.58	ctrl
5.18	ctrl
6.11	ctrl
4.50	ctrl
4.61	ctrl

```
ggplot(PlantGrowth, aes(x=group, y=weight)) +
  geom_boxplot() +
  ggtitle("Plant growth with\ndifferent treatments")+
  xlab("this is xlab")+
  ylab("this is ylab")
```

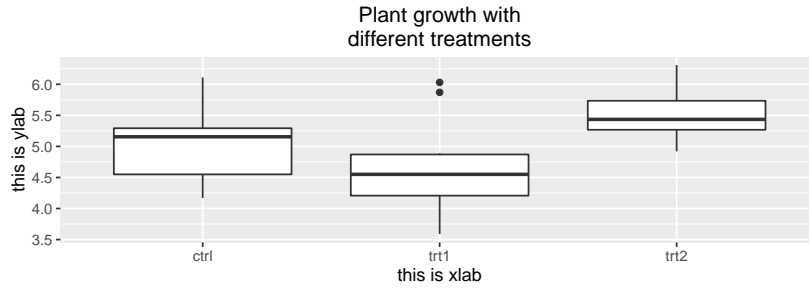
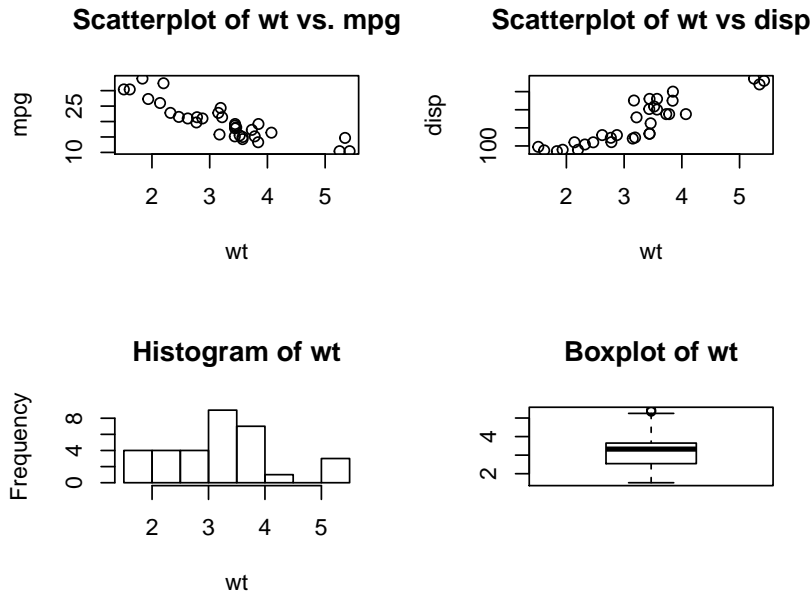


图 42: ggplot2 包里如何添加图片标题

如何在一个画布里放多个图片

在 `graphics` 包里设置画布多图非常简单, 用 `par()` 函数里的 `mfrow` 参数。如果想 2 行 2 列, 就是 `mfrow=c(2,2)`; 如果想 3 行 1 列, 就用 `mfrow=c(3,1)`。

```
attach(mtcars)
par(mfrow=c(2,2))
plot(wt,mpg, main="Scatterplot of wt vs. mpg")
plot(wt,disp, main="Scatterplot of wt vs disp")
hist(wt, main="Histogram of wt")
boxplot(wt, main="Boxplot of wt")
```

图 43: `par()` 函数 `mfrow` 设置多个图片同个画布

如果想第一行 1 个图, 第二行 2 个图, 就要用到 `layout()` 函数来设置⁸。

⁸ 参考 Combining Plots

```

attach(mtcars)
## matrix(c(1,1,2,3), 2, 2, byrow = TRUE) 指出了三张图的具体位置
##      [,1] [,2]
## [1,]    1    1
## [2,]    2    3
## widths 设置两列的比例
## heights 控制两行的比例
layout(matrix(c(1,1,2,3), 2, 2, byrow = TRUE),
        widths=c(3,1), heights=c(1,2))

hist(wt)
hist(mpg)
hist(dis)

```

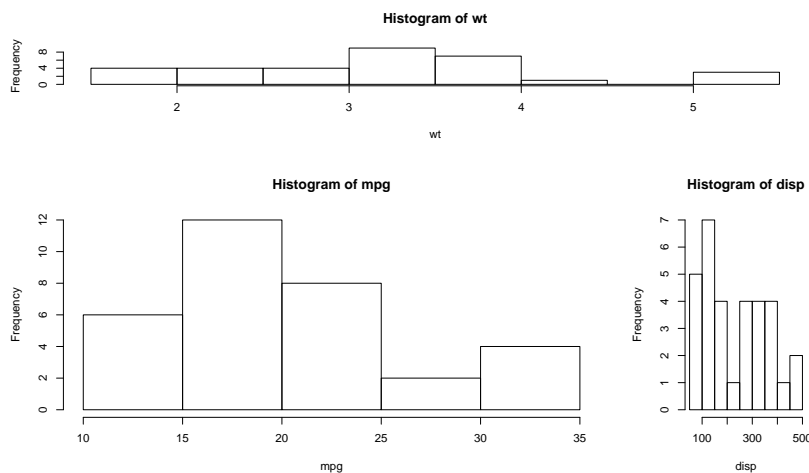


图 44: layout() 设置多个图片同个画布

在 ggplot2 里需要用其他的附加包 (cowplot 或者 gridExtra) 来完成这个工作, 个人推荐用后者, 因为比较完善。对于用这两个包在同一个画布里画多个图像的全面介绍可以参考《Easy way to mix multiple graphs on the same page》。

下面仅展示 2 个例子:

1. 用 cowplot::ggdraw() 设置多个图片同个画布

```

library(cowplot)
sp <- ggplot(mtcars, aes(x = mpg, y = hp, colour = factor(cyl)))+
  geom_point(size=2.5)
bp <- ggplot(diamonds, aes(clarity, fill = cut)) +
  geom_bar() +
  theme(axis.text.x = element_text(angle=90, vjust=0.5))
plot.iris <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) +

```

```

geom_point() + facet_grid(. ~ Species) + stat_smooth(method = "lm") +
background_grid(major = 'y', minor = "none") +
panel_border()

# 画等行等列图像
# plot_grid(sp, bp, labels=c("A", "B"), ncol = 2, nrow = 1)
# 画任意位置图像
ggdraw() +
  draw_plot(plot.iris, 0, .5, 1, .5) +
  draw_plot(sp, 0, 0, .5, .5) +
  draw_plot(bp, .5, 0, .5, .5) +
  draw_plot_label(c("A", "B", "C"), c(0, 0, 0.5), c(1, 0.5, 0.5), size = 15)

```

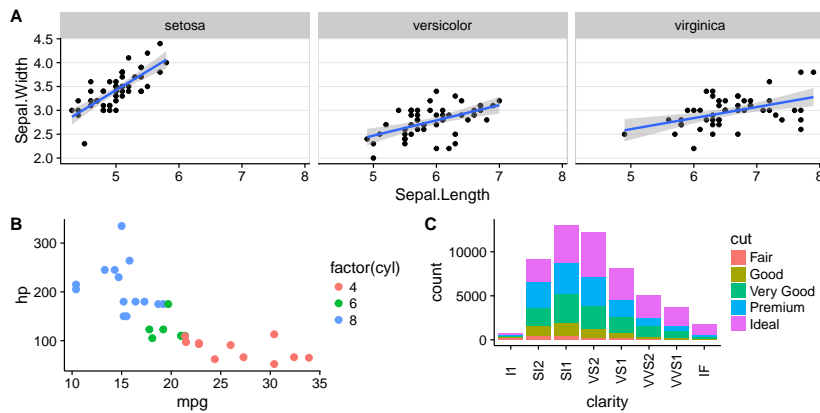


图 45: cowlplot::ggdraw() 设置多个图片同个画布

2.gridExtra::grid.arrange() 设置多个图片同个画布

在这里我们使用一个维生素 D 对豚鼠牙齿生长的影响的数据 (ToothGrowth), 该数据记录了维生素 D 含量同豚鼠牙齿长度的关系。

表 12: PlantGrowth 数据

len	supp	dose
4.2	VC	0.5
11.5	VC	0.5
7.3	VC	0.5
5.8	VC	0.5
6.4	VC	0.5
10.0	VC	0.5

```

df <- ToothGrowth
df$dose <- as.factor(df$dose)
## 计量同牙齿长度的箱线图

```

```

bp <- ggplot(df, aes(x=dose, y=len, color=dose)) +
  geom_boxplot() +
  theme(legend.position = "none")
## 计量同牙齿长度的 Cleveland 点图
dp <- ggplot(df, aes(x=dose, y=len, fill=dose)) +
  geom_dotplot(binaxis='y', stackdir='center')+
  stat_summary(fun.data=mean_sdl, fun.args = list(mult=1),
              geom="pointrange", color="red")+
  theme(legend.position = "none")
## 计量同牙齿长度的提琴图
vp <- ggplot(df, aes(x=dose, y=len)) +
  geom_violin()+
  geom_boxplot(width=0.1)
## 计量同牙齿长度的散点图 (jitter 抖动模式)
sc <- ggplot(df, aes(x=dose, y=len, color=dose, shape=dose)) +
  geom_jitter(position=position_jitter(0.2))+
  theme(legend.position = "none") +
  theme_gray()

library(gridExtra)
grid.arrange(bp, dp, vp, sc, ncol=2, nrow =2)

## Warning: Computation failed in `stat_summary()``:
## Hmisc package required for this function

```

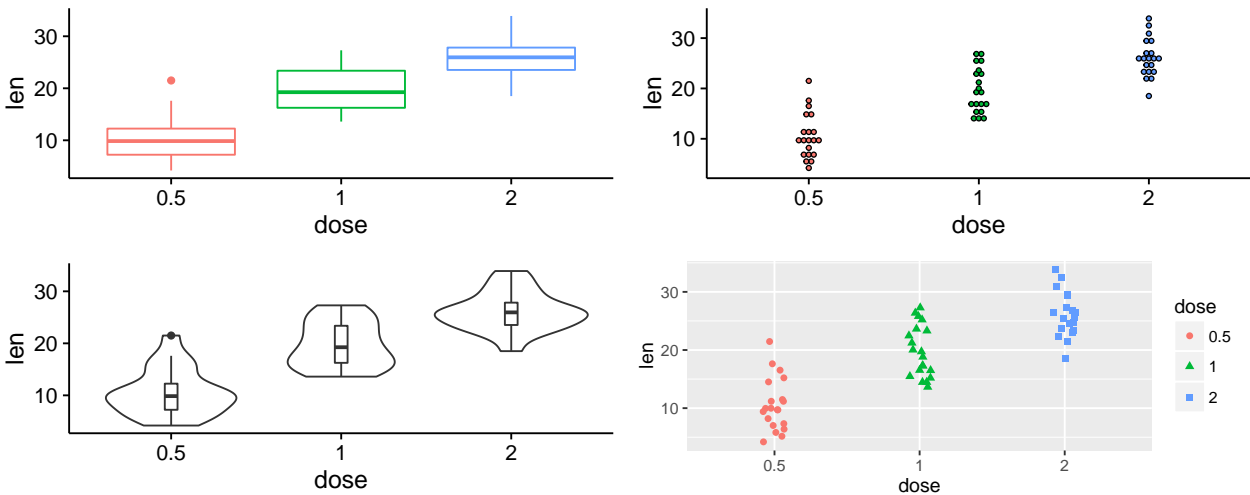


图 46: gridExtra::grid.arrange() 设置多个图片同个画布

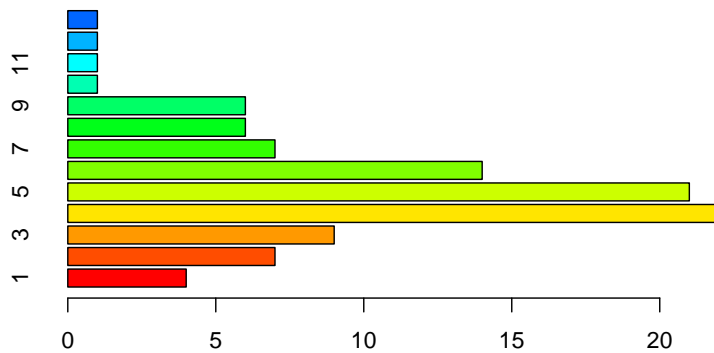
如何旋转条形图坐标轴

在散点图和折线图里都是 x 和 y 的变量对掉就等于旋转了 x 轴和 y 轴。但是一般遇到这个问题，都是想把条形图转个 90 度。

在 `graphics::barplot()` 里专门有个参数是做这件事 `horiz=TRUE`，在 `graphics::boxplot()` 里这个参数是 `horizontal=TRUE`。

```
tN <- table(Ni <- stats::rpois(100, lambda = 5))
barplot(tN, col = rainbow(20), horiz=TRUE)
```

图 47: `barplot()` 水平显示条形图



在 `ggplot2` 包里用 `coord_flip()` 函数来完成这个工作（所有图形都可以用这个函数来翻转）。

```
ggplot(PlantGrowth, aes(x=group, y=weight))+
  geom_boxplot() + coord_flip()
```

`ggplot2` 的背景怎么去掉

`ggplot2` 包里自带了 9 中主题，分别是：

```
theme_grey(base_size = 11, base_family = "") # 默认字号为 11pt
theme_gray(base_size = 11, base_family = "")
theme_bw(base_size = 12, base_family = "") # 默认字号为 12pt
theme_linedraw(base_size = 12, base_family = "")
theme_light(base_size = 12, base_family = "")
theme_minimal(base_size = 12, base_family = "")
theme_classic(base_size = 12, base_family = "")
theme_dark(base_size = 12, base_family = "")
theme_void(base_size = 12, base_family = "")
```

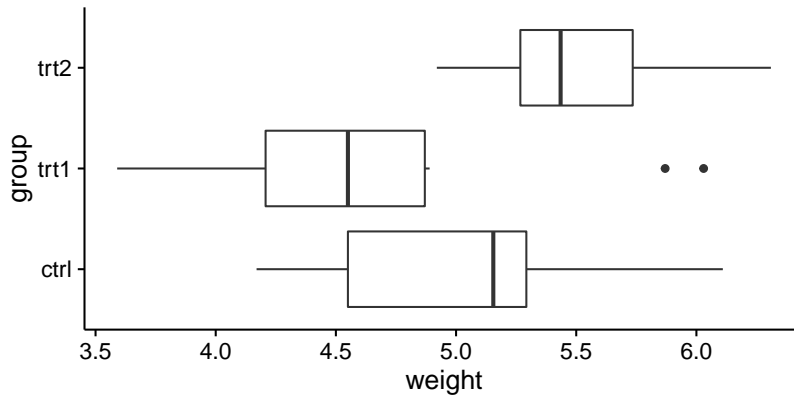


图 48: coord_flip() 水平显示直方图

下面是这 9 中主题的展示。注意在 `ggplot2` 2.1.0 这个版本中 `theme_classic()` 有 bug⁹，导致图片无法正常显示坐标轴。

在修改后的版本没有推送到 CRAN 之前，可以试着安装使用开发版 `devtools::install_github("hadley/ggplot2")`（但我测试没有成功）。

经典灰色主题背景

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +
  geom_boxplot() +
  theme_grey()
```

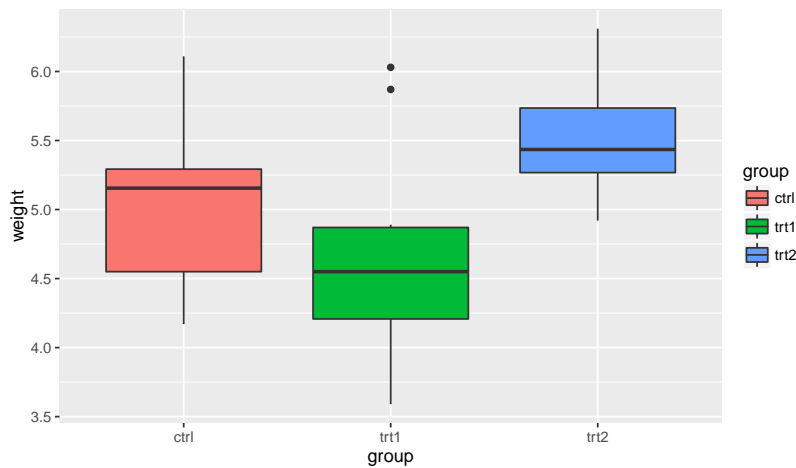


图 49: theme_grey() 背景

经典灰色主题背景（美式拼写）

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +
  geom_boxplot() +
  theme_gray()
```

⁹ `theme_classic()` no axis lines in version 2.1

经典黑白主题背景

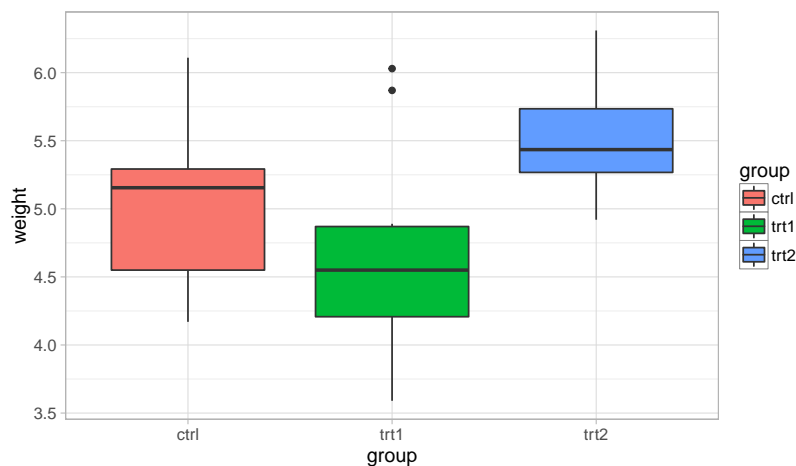
```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +  
  geom_boxplot() +  
  theme_bw()
```

黑色线条，浅灰色背景线主题

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +  
  geom_boxplot() +  
  theme_linedraw()
```

浅色背景线主题

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +  
  geom_boxplot() +  
  theme_light()
```



最简风格主题

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +  
  geom_boxplot() +  
  theme_minimal()
```

经典只有坐标轴有颜色的主题（目前因包内 bug 显示错误）

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +  
  geom_boxplot() +  
  theme_classic()
```

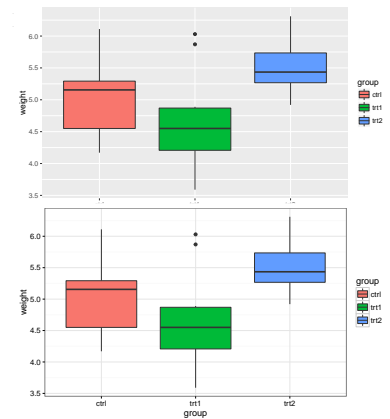


图 51: theme_bw() 背景

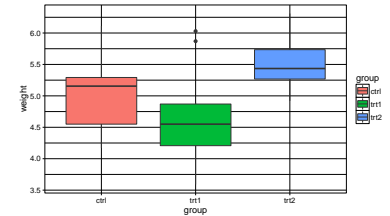


图 52: theme_linedraw() 背景

图 53: theme_light() 背景

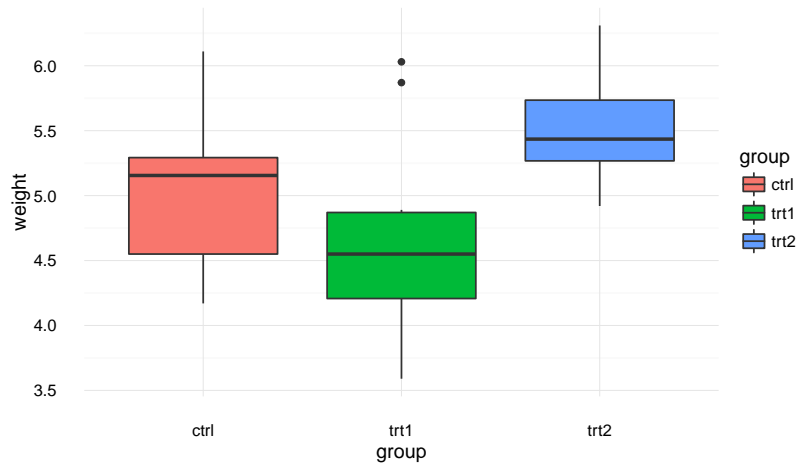


图 54: ggplot2 包里如何更改背景

深灰色主题

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +
  geom_boxplot() +
  theme_dark()
```

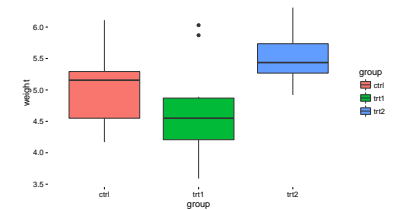
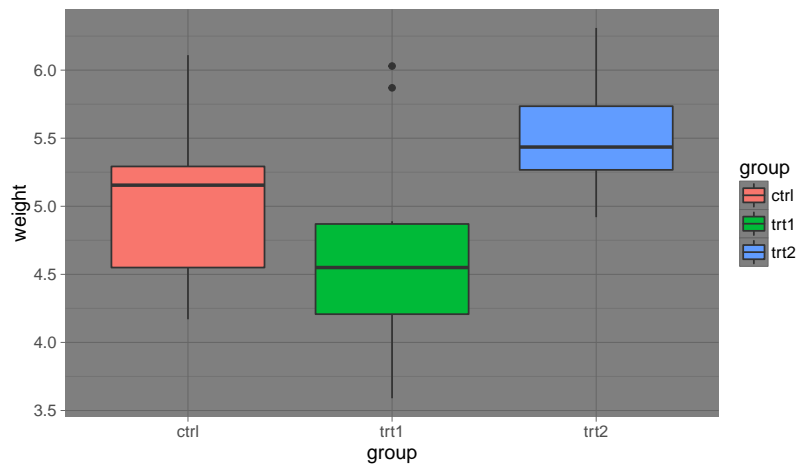


图 55: theme_classic() 背景

图 56: theme_dark() 背景

完全没有背景的主题

```
ggplot(PlantGrowth, aes(x=group, y=weight, fill=group)) +
  geom_boxplot() +
  theme_void()
```

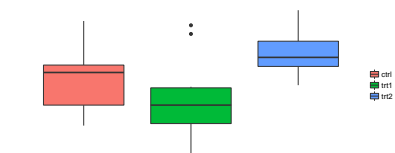


图 57: theme_void() 背景

如果只想要普通坐标轴有颜色，没有背景的图片，请参考下面这段代码的设置，这段代码是 Stackoverflow 上的经典问答，从中我们可以了解到 ggplot2 的版本经历了多少变化¹⁰。

在 ggplot2 2.0.0 版本里可以使用 `+ theme_classic()` 来完成，但是在更高的版本中由于有 bug，所以这个主题基本没用了（所以我上图中的 `theme_classic()` 主题也显示的不是它本来的样子）。

在高版本中的一个补救措施如下图所示。

```
library(ggplot2)
a <- seq(1,20)
b <- a^0.25
df <- as.data.frame(cbind(a,b))
ggplot(df, aes(a, b)) +
  geom_point() +
  theme(axis.line.x = element_line(color = "black"),
        axis.line.y = element_line(color = "black"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.border = element_blank(),
        panel.background = element_blank())
```

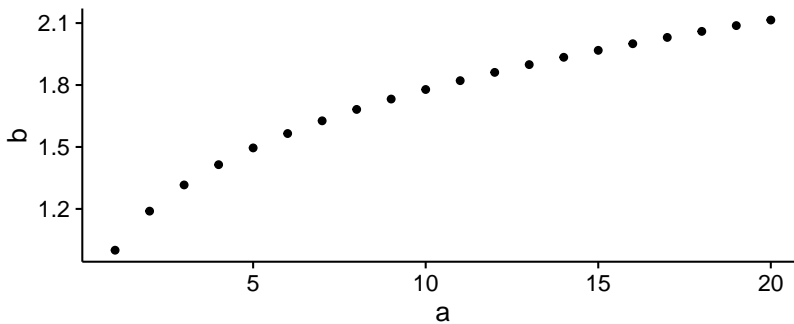


图 58: 去掉背景仅显示坐标轴

如果还想了解更多的 ggplot2 背景，请参考 `ggthemes` 包¹¹。

¹¹ Introduction to ggthemes