6.2　饼图

饼图在商业世界中无所不在，然而多数统计学家，包括相应R文档的编写者却都对它持否定态度。相对于饼图，他们更推荐使用条形图或点图，因为相对于面积，人们对长度的判断更精确。也许由于这个原因，R中饼图的选项与其他统计软件相比十分有限。

饼图可由以下函数创建：

```
pie(x, labels)
```

其中*x*是一个非负数值向量，表示每个扇形的面积，而*labels*则是表示各扇形标签的字符型向量。代码清单6-5给出了四个示例，结果如图6-6所示。

代码清单6-5　饼图

```
par(mfrow=c(2, 2))
slices <- c(10, 12,4, 16, 8)
lbls <- c("US", "UK", "Australia", "Germany", "France")

pie(slices, labels = lbls,
    main="Simple Pie Chart")

pct <- round(slices/sum(slices)*100)
lbls2 <- paste(lbls, " ", pct, "%", sep="")
pie(slices, labels=lbls2, col=rainbow(length(lbls2)),
    main="Pie Chart with Percentages")

library(plotrix)
pie3D(slices, labels=lbls,explode=0.1,
      main="3D Pie Chart ")

mytable <- table(state.region)
lbls3 <- paste(names(mytable), "\n", mytable, sep="")
pie(mytable, labels = lbls3,
    main="Pie Chart from a Table\n (with sample sizes)")
```

❶ 将四幅图形组合为一幅

❷ 为饼图添加比例数值

❸ 从表格创建饼图

首先，你做了图形设置，这样四幅图形就会被组合为一幅❶。（多幅图形的组合在第3章中介绍过。）然后，你输入了前三幅图形将会使用的数据。

对于第二幅饼图❷，你将样本数转换为比例值，并将这项信息添加到了各扇形的标签上。如第 3 章所述，第二幅饼图使用rainbow()函数定义了各扇形的颜色。这里的rainbow(length(lbls2))将被解析为rainbow(5)，即为图形提供了五种颜色。

第三幅是使用plotrix包中的pie3D()函数创建的三维饼图。请在第一次使用之前先下载并安装这个包。如果说统计学家们只是不喜欢饼图的话，那么他们对三维饼图的态度就一定是唾弃了（即使他们私下感觉三维饼图好看）。这是因为三维效果无法增进对数据的理解，并且被认为是分散注意力的视觉花瓶。

第四幅图演示了如何从表格创建饼图❸。在本例中，你计算了美国不同地区的州数，并在绘制图形之前将此信息附加到了标签上。

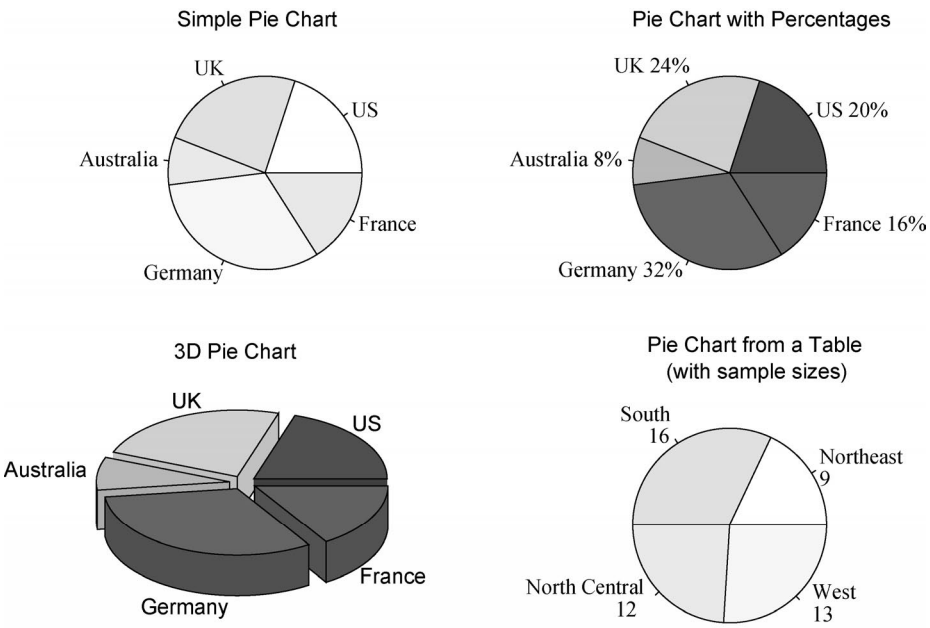

图6-6 饼图示例

饼图让比较各扇形的值变得困难（除非这些值被附加在标签上）。例如，观察（第一幅）最简单的饼图，你能分辨出美国（US）和德国（Germany）的大小吗？（如果你可以，说明你的洞察力比我好。)为改善这种状况,我们创造了一种称为扇形图(fan plot)的饼图变种。扇形图(Lemon & Tyagi，2009）为用户提供了一种同时展示相对数量和相互差异的方法。在R中，扇形图是通过plotrix包中的fan.plot()函数实现的。

考虑以下代码和结果图（图6-7）：

```
library(plotrix)
slices <- c(10, 12,4, 16, 8)
lbls <- c("US", "UK", "Australia", "Germany", "France")
fan.plot(slices, labels = lbls, main="Fan Plot")
```

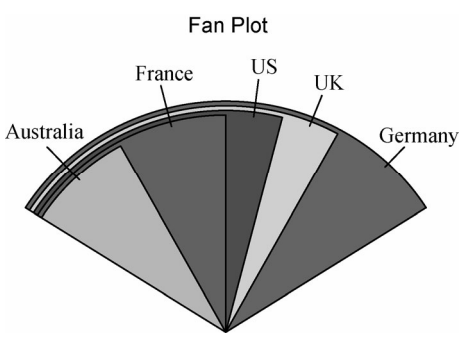

图6-7 国别数据的扇形图

在一幅扇形图中，各个扇形相互叠加，并对半径做了修改，这样所有扇形就都是可见的。在这里可见德国对应的扇形是最大的，而美国的扇形大小约为其60%。法国的扇形大小似乎是德国的一半，是澳大利亚的两倍。请记住，在这里扇形的宽度（*width*）是重要的，而半径并不重要。

如你所见，确定扇形图中扇形的相对大小比饼图要简单得多。扇形图虽然尚未普及，但它仍然是新生力量。既然已经讲完了饼图和扇形图，就让我们转到直方图上吧。与条形图和饼图不同，直方图描述的是连续型变量的分布。

6.3　直方图

直方图通过在*X*轴上将值域分割为一定数量的组，在*Y*轴上显示相应值的频数，展示了连续型变量的分布。可以使用如下函数创建直方图：

```
hist(x)
```

其中的*x*是一个由数据值组成的数值向量。参数`freq=FALSE`表示根据概率密度而不是频数绘制图形。参数`breaks`用于控制组的数量。在定义直方图中的单元时，默认将生成等距切分。代码清单6-6提供了绘制四种直方图的代码，绘制结果见图6-8。

代码清单6-6　直方图

```
par(mfrow=c(2,2))

hist(mtcars$mpg)                                    ❶ 简单直方图

hist(mtcars$mpg,
     breaks=12,                                     ❷ 指定组数和颜色
     col="red",
     xlab="Miles Per Gallon",
     main="Colored histogram with 12 bins")

hist(mtcars$mpg,                                    ❸ 添加轴须图
     freq=FALSE,
     breaks=12,
     col="red",
     xlab="Miles Per Gallon",
     main="Histogram, rug plot, density curve")
rug(jitter(mtcars$mpg))
lines(density(mtcars$mpg), col="blue", lwd=2)

x <- mtcars$mpg                                     ❹ 添加正态密度曲线和外框
h<-hist(x,
        breaks=12,
        col="red",
        xlab="Miles Per Gallon",
        main="Histogram with normal curve and box")
xfit<-seq(min(x), max(x), length=40)
yfit<-dnorm(xfit, mean=mean(x), sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
box()
```

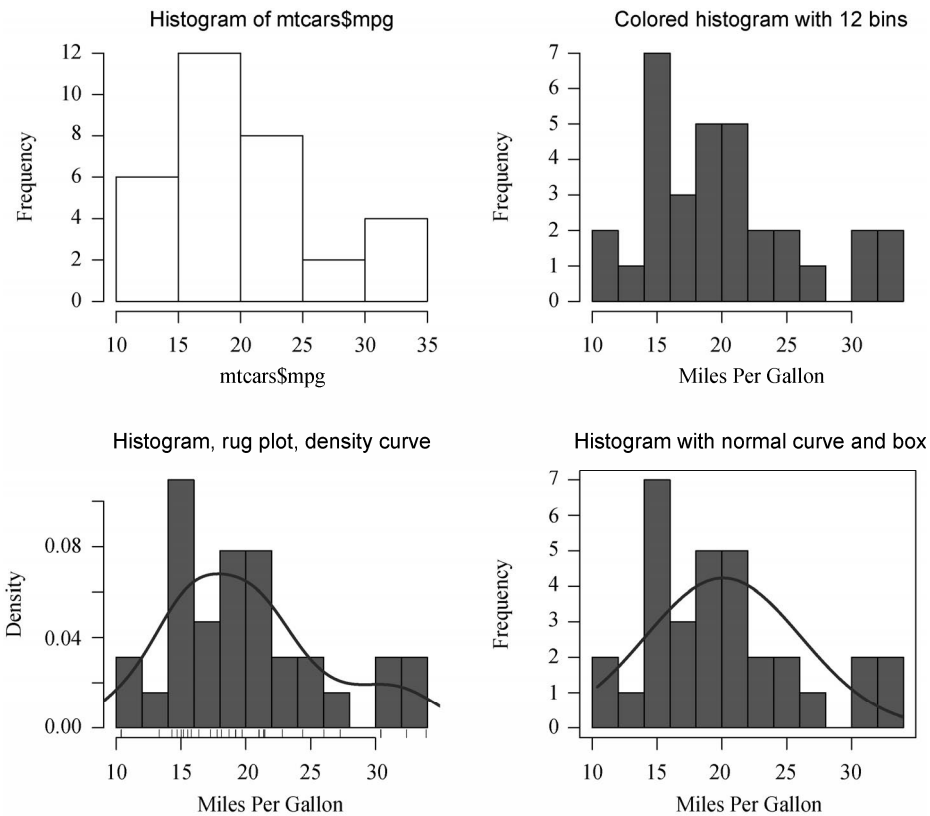

图6-8 直方图示例

第一幅直方图❶展示了未指定任何选项时的默认图形。这个例子共创建了五个组，并且显示了默认的标题和坐标轴标签。对于第二幅直方图❷，我们将组数指定为12，使用红色填充条形，并添加了更吸引人、更具信息量的标签和标题。

第三幅直方图❸保留了上一幅图中的颜色、组数、标签和标题设置，又叠加了一条密度曲线和轴须图（rug plot）。这条密度曲线是一个核密度估计，会在下节中描述。它为数据的分布提供了一种更加平滑的描述。我们使用`lines()`函数叠加了这条蓝色、双倍默认线条宽度的曲线。最后，轴须图是实际数据值的一种一维呈现方式。如果数据中有许多结[①]，你可以使用如下代码将轴须图的数据打散：

```
rug(jitter(mtcars$mpag, amount=0.01))
```

这样将向每个数据点添加一个小的随机值（一个`±amount`之间的均匀分布随机数），以避免重叠的点产生影响。

第四幅直方图❹与第二幅类似，只是拥有一条叠加在上面的正态曲线和一个将图形围绕起来

① 数据中出现相同的值，称为结（tie）。——译者注

的盒型。用于叠加正态曲线的代码来源于R-help邮件列表上由Peter Dalgaard发表的建议。盒型是使用box()函数生成的。

6.4　核密度图

在上节中，你看到了直方图上叠加的核密度图。用术语来说，核密度估计是用于估计随机变量概率密度函数的一种非参数方法。虽然其数学细节已经超出了本书的范畴，但从总体上讲，核密度图不失为一种用来观察连续型变量分布的有效方法。绘制密度图的方法（不叠加到另一幅图上方）为：

```
Plot(density(x))
```

其中的x是一个数值型向量。由于plot()函数会创建一幅新的图形，所以要向一幅已经存在的图形上叠加一条密度曲线，可以使用lines()函数（如代码清单6-6所示）。

代码清单6-7给出了两幅核密度图示例，结果如图6-9所示。

代码清单6-7　核密度图

```
par(mfrow=c(2,1))
d <- density(mtcars$mpg)

plot(d)

d <- density(mtcars$mpg)
plot(d, main="Kernel Density of Miles Per Gallon")
polygon(d, col="red", border="blue")
rug(mtcars$mpg, col="brown")
```

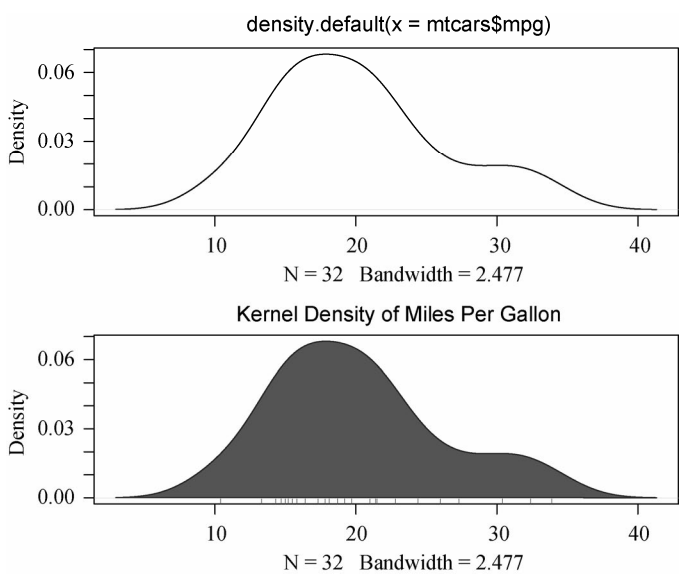

图6-9　核密度图

在第一幅图中，你看到的是完全使用默认设置创建的最简图形。在第二幅图中，你添加了一个标题，将曲线修改为蓝色，使用实心红色填充了曲线下方的区域，并添加了棕色的轴须图。polygon()函数根据顶点的x和y坐标（本例中由density()函数提供）绘制了多边形。

核密度图可用于比较组间差异。可能是由于普遍缺乏方便好用的软件，这种方法其实完全没有被充分利用。幸运的是，sm包漂亮地填补了这一缺口。

使用sm包中的sm.density.compare()函数可向图形叠加两组或更多的核密度图。使用格式为：

```
sm.density.compare(x, factor)
```

其中的x是一个数值型向量，factor是一个分组变量。请在第一次使用sm包之前先安装它。代码清单6-8中提供了一个示例，它比较了拥有4个、6个或8个汽缸车型的每加仑汽油行驶英里数。

代码清单6-8 可比较的核密度图

```
par(lwd=2)                                                    ❶ 双倍线条宽度
library(sm)
attach(mtcars)

cyl.f <- factor(cyl, levels= c(4,6,8),                        ❷ 创建分组因子
              labels = c("4 cylinder", "6 cylinder",
                         "8 cylinder"))

sm.density.compare(mpg, cyl, xlab="Miles Per Gallon")         ❸ 绘制密度图
title(main="MPG Distribution by Car Cylinders")

colfill<-c(2:(1+length(levels(cyl.f))))                       ❹ 通过鼠标单击添加图例
legend(locator(1), levels(cyl.f), fill=colfill)

detach(mtcars)
```

par()函数将所绘制的线条设置为双倍宽度（lwd=2），这样它们在书中就会更易读❶。接下来载入了sm包，并绑定了数据框mtcars。

在数据框mtcars ❷中，变量cyl是一个以4、6或8编码的数值型变量。为了向图形提供值的标签，这里cyl转换为名为cyl.f的因子。函数sm.density.compare()创建了图形❸，一条title()语句添加了主标题。

最后，添加了一个图例以增加可解释性❹。（图例已在第3章介绍。）首先创建的是一个颜色向量，这里的colfill值为c(2, 3, 4)。然后通过legend()函数向图形上添加一个图例。第一个参数值locator(1)表示用鼠标点击想让图例出现的位置来交互式地放置这个图例。第二个参数值则是由标签组成的字符向量。第三个参数值使用向量colfill为cyl.f的每一个水平指定了一种颜色。结果如图6-10所示。

如你所见，核密度图的叠加不失为一种在某个结果变量上跨组比较观测的强大方法。你可以看到不同组所含值的分布形状，以及不同组之间的重叠程度。（这段话的寓意是，我的下一辆车将是四缸的——或是一辆电动的。）

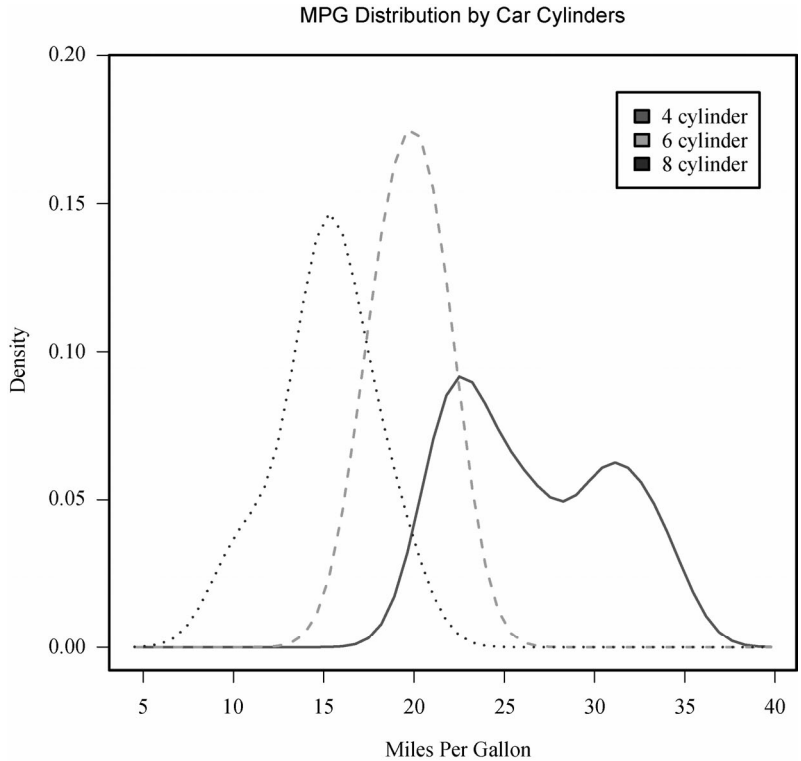

图6-10　按汽缸个数划分的各车型每加仑汽油行驶英里数的核密度图

箱线图同样是一项用来可视化分布和组间差异的绝佳图形手段（并且更常用），我们接下来讨论它。

6.5　箱线图

箱线图（又称盒须图）通过绘制连续型变量的五数总括，即最小值、下四分位数（第25百分位数）、中位数（第50百分位数）、上四分位数（第75百分位数）以及最大值，描述了连续型变量的分布。箱线图能够显示出可能为离群点（范围 ± 1.5*IQR以外的值，IQR表示四分位距，即上四分位数与下四分位数的差值）的观测。例如：

```
boxplot(mtcars$mpg, main="Box plot", ylab="Miles per Gallon")
```

生成了如图6-11所示的图形。为了图解各个组成部分，我手工添加了标注。

默认情况下，两条须的延伸极限不会超过盒型各端加1.5倍四分位距的范围。此范围以外的值将以点来表示（在这里没有画出）。

举例来说，在我们的车型样本中，每加仑汽油行驶英里数的中位数是19.2，50%的值都落在了15.3和22.8之间，最小值为10.4，最大值为33.9。我是如何从图中如此精确地读出了这些值呢？