Introduction

ggplot2采用了图层的设计方式，你可以从原始的图层开始，首先绘制原始数据，然后不断地添加图形注释和统计汇总结果

qplot

qplot(carat, price, data=diamonds, alpha=I(1/10))

alpha取值从0(完全透明)到1(完全不透明)

使用I()来设置图形属性，colour=I("red"), size=I(2)

二维分布:

geom="point"绘制散点图，当指定x和y时qplot的默认设置

geom="smooth"将拟合一条平滑曲线，并将曲线和标准误展示在图中

geom="boxplot"绘制箱线胡须图，用以概括一系列点点分布情况

geom="path"和geom="line"可以在数据点之间绘制连线，探索时间和其他变量之间关系

一维分布:

连续变量，geom="histogram"绘制直方图，geom="freqpoly"绘制频率多边形，geom="density"绘制密度曲线，当仅有x参数时，默认为直方图

离散变量，geom="bar"绘制条形图, 没有binwidth参数, histogram有

使用smooth添加平滑曲线，se=F,取消标准误，method="loess"选择平滑器；**当n较小时，默认"loess"，使用局部回归的方法**；通过span来控制平滑程度，范围从0(很不平滑)到1(很平滑)

qplot(carat, price, data=dsmall, geom=c("point","smooth"),span=0.2)

method="gam",formula=y~s(x)调用mgvc包拟合广义可加模型

library(mgcv); qplot(carat, price, data=dsmall,
geom=c("point","smooth"),method="gam",formula=y~s(x))

methon="lm", 拟合线性模型。默认条件拟合一条直线，通过指定formula=y~poly(x,2) 拟合一个二次多项式或加载splines包以使用自然样条:formula=y~ns(x,2)

method="rlm", 与lm类似，但是采用了一种更稳定的拟合算法，使得结果对异常值不太敏感。

扰动图: qplot(color, price/carat, data=diamonds, **geom="jitter"**, alpha=I(1/5))

箱线图: qplot(color, carat/price, data=diamonds, **geom="boxplot"**,color=color)

直方图: qplot(carat,data=diamonds, **geom="histogram"**, fill=color, binwithd=0.3, xlim=c(0,3))

密度图: qplot(carat, data=diamonds, geom="density", colour=color)

当一个分类变量被映射到某个图形属性上，几何对象会自动按这个变量进行拆分

直方图: qplot(color, data=diamonds, geom="bar", weight=carat) + scale_y_continuous("carat") 使用weight=carat进行加权

线条图和路径图常用于可视化时间序列数据，线条图将点从左到右进行连接，而路径图则按照点在数据集中的顺序对其进行连接(线条图就等价于将数据按照x取值进行排序，然后绘制路径图)

qplot(date, unemploy/pop, data, data=economics, geom="line")

year <- function(x)as.POSIXlt(x)$year + 1900

qplot(unemploy/pop, uempmed, data=economics, geom="path", colour=year(date))

分面: 将数据分割成若干子集，然后创建一个图形矩阵，将每一个子集绘制到图形矩阵的窗格中

row_var ~ col_var; **.~row_var**创建单列多行的图形矩阵; **..density..** 映射密度二不是频数到y轴

qplot(carat, data=diamonds, facets=color~., geom="histogram", binwidth=0.1, xlim=c(0,3))

qplot(carat,**..density..**, data=daimongs, facets=color~., geom="histogram", binwidth=0.1, xlim=c(0,3))

xlim,ylim: 设置x轴和y轴的显示区间

log: 对坐标轴取对数，log="x", log="xy"

main: 图形的主标题，main="plot title"

xlab,ylab: 设置x轴和y轴的标签文字，和主标题一样，这两个参数的取值可以是字符串或数学表达式

aes(横坐标, 纵坐标, 大小, 颜色, 形状), 同时图形属性也意味着分组

点(point), 线(lines), 条(bar)都是几何对象的具体形式，被称作geom，几何对象决定了图形的"类型"(type)

ggplot(mpg,aes(displ,hwy,color=factor(cyl)))+geom_point()+geom_smooth(method="lm")

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(默认的图形属性映射), 图层，标度，坐标和分面的列表, print()函数将其呈现到屏幕上; ggsave()函数将其保存到磁盘; summary()简单查看它的结构; 使用save()函数把它的缓存副本保存到磁盘, 这样可以保存一个图形对象的完整副本，你可以调用load()函数来重现该图。

save(p, file="plot.rdata"); load("plot.rdata"); ggsave("plot.png", with=5, height=5)

ggsave(): path设定图形存储的路径, 它会根据文件扩展名自动选择正确的图形设备; 三个控制输出尺寸的参数, 若空白, 则使用当前屏幕图形设备尺寸, width和height设置绝对尺寸大小, scale设置图像相对屏幕展示的尺寸; 对于光栅图形, dpi参数控制图形的分辨率, 默认值为300, 适合大部分打印设备, 修改为600用于高分辨率输出

数据，必须是一个数据框(data frame); 一组图像映射; 几何对象, 统计变换, 对原数据做一些有用的统计变换; 位置调整, 通过元素位置来避免图形重合

%+%: 添加新的数据集以代替原来的数据集

p <- ggplot(mtcars, aes(mpg, wt, colour=cyl))+geom_point()

mtcars <- transform(mtcars, mpg=mpg^2)

p %+% mtcars

aes()函数里的变量都必须包含于默认数据集或者图层数据集中，这是保证ggplot2对象都是自含型的重要方式之一，方便存储和重复使用; aes()可以对图层进行添加, 修改和删除默认映射

aes(colour=cyl) aes(mpg, wt, colour=cyl) 添加

aes(y=disp) aes(mpg, disp) 修改

aes(y=NULL) aes(mpg) 删除

可在图层的参数中将图像属性设置为单一值(colour="red")

p + geom_point(colour="darkblue")将点颜色设置为深蓝色

p + geom_point(aes(colour="darkblue"))这里将colour**映射**到"darkblue"颜色，实际上先创建了一个只含有"darkblue"字符的变量, 然后将colour映射到这个新变量, 又因为data中没有这个object, 则将其看作普通字符串，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(group): 图中所有离散型变量的交互作用被设为分组的默认值

ggplot(Oxboys, aes(age, height, group=Subject))+geom_line() 未分组等价于group=1

修改默认分组，如果图像中含有离散型变量，可以采用绘制交互作用图, 轮廓图以及平行坐标图时采用策略绘制连接所有分组的线条

boysbox <- ggplot(Oxboys, aes(Occasion, height))+geom_boxplot(), 此时Occassion是一个离散型变量，所以默认分组就是Occassion，可使用aes(group=Subject)修改第一层默认分组

boysbox + geom_line(aes(group=Subject), colour="#3366FF") or

boysbox + geom_line(aes(group=Subject), color=Subject))

几何图形对象, 简称geom， 它执行着图层的实际渲染, 控制着生成的图像类型, 每个几何对象都有一组它能识别的图形属性和一组绘图所需的值，例如一个点含有的颜色, 大小和形状等图形属性, 以及x和y位置坐标。

aline 线, 由斜率和截距决定geom_abline(slope, intercept, na.rm = FALSE, show.legend = NA); aes(colour, linetype, size)

hline 水平线geom_hline(yintercept, na.rm = FALSE, show.legend = NA); aes(colour, linetype, size)

jitter 给点添加扰动，减轻图形重叠问题shortcut: geom_point(position="jitter"); aes(colur, fill, shape, size, x, y)

quantile 添加分位回归线geom_quantile(); aes(colur, linetype, size, weight, x, y)

smooth 添加光滑的条件均值

vline 竖直线geom_vline(xintercept, na.rm=FALSE, show.legend=NA); aes(colour, linetype, size)

统计变换, stat, 即对数据进行统计变换, 它通常以某种方式对数据信息进行汇总: count, density, x

ggplot(diamonds, aes(carat))+stat_bin(aes(y..count.., color=cut), binwidth=0.1, geom="line", position="jitter")

count, 每个组里观测值的数目

density, 每个组里观测值的密度(占整体的百分数/组宽), 基本相当于count除以count的总数

x, 组的中心位置

这些生成变量(generated variable)可以被直接调用，例如直方图默认条形高度为观测值的count，可以使用density来代替

ggplot(diamonds, aes(carat))+geom_histogram(aes(y=..density..), binwidth=0.1)

ggplot(diamonds, aes(carat))+geom_histogram()+stat_bin(aes(y=..count..), binwidth=0.1)

生成变量的名字必须用..围起来，这样可以防止原数据中的变量和生成变量重名时造成混淆，并且以后处理代码时，可以很清晰得分辨出哪些变量时由统计变换生成的

dodge 避免重叠，并排放置

fill 堆叠图形元素并将高度标准化为1

jitter 给点添加扰动避免重合

stack 将图形元素堆叠起来

ggplot(diamonds, aes(clarity))+geom_bar(aes(fill=cut),position="stack")

ggplot(diamonds, aes(clarity))+geom_bar(aes(fill=cut),position="dodge")

ggplot(diamonds,aes(clarity))+geom_bar(aes(fill=cut))

p63右图无法绘制

二维几何对象, x和y是两种不可或缺的图形属性，同时可以接受colour和size图形属性，针对填充型几何对象(条形, 瓦片(tile)和多边形)还可以接受fill图形属性, 点使用shape图形属性, 线和路径接受linetype图形属性, geom_text()可在指定点处添加标签，它是几何对象中唯一一个需要额外图形属性的:它需要制定label参数，还可以设置hjust和vjust控制文本的横纵位置，使用angle来控制文本的旋转。

geom_boxplot=stat_boxplot + geom_boxplot

箱线图对连续变量取条件, 先对数据预先巧妙封箱(binning)处理：

library(plyr); qplot(carat, depth, data=diamonds, geom="boxplot", **group=round_any(carat, 0.1, floor)**, xlim=c(0,3))

group=round_any(carat, 0.1, floor)来获得对变量carat以0.1个单位为大小封箱后的箱线图

geom_jitter=position_jitter + geom_point: 通过在离散型分布上添加随机噪音以避免遮盖绘制问题

qplot(class, dev, data=mpg, geom="jitter")

geom_density=stat_density + geom_area: 基于核平滑方法进行平滑后得到的频率多边形

qplot(depth, data=diamonds, geom="density", xlim=c(54,70))

添加图形注解有两种基本的方式: 逐个添加或批量添加

unemp <- qplot(date, unemploye, data=economics, geom="line", **xlab="", ylab="No. unemployed(1000s)"**)

unemp + geom_vline(aes(xintercept=as.numeric(start)), data=presidential)

unemp + geom_rect(aes(NULL, NULL, xmin=start, xmax=end, fill=party), ymin=yrng[1], ymax=yrng[2], data=presidential, alpha=0.2) + scale_fill_manual(values=c("blue","red"))

geom_text: 可添加文字叙述或为点添加标签

geom_vline, geom_hline: 向图形添加垂直线或水平线

geom_abline: 向图形添加任意斜率和截距的直线

geom_rect: 可强调图形中感兴趣的矩形区域, 拥有xmin, xmax, ymin和ymax几种图形属性

geom_line, geom_path, geom_segment: 添加直线, 因此这些几何对象都有一个arrow参数, 可以用来在线上放置一个箭头, 使用arrow()函数绘制箭头, 它拥有angle, length, ends和type几个参数

含权数据: 打个比方说, 一件事情, 你给它打100分, 你的老板给它打60分, 如果平均, 则是(100+60)/2=80分。但因为老板说的话分量比你重, 假如老板的权重是2, 你是1, 这时求平均值就是加权平均了, 结果是(100x1 + 60x2)/(1+2)=73.3分, 显然向你的老板那里倾斜了。假如老板权重是3，你的权重是1，结果是（100x1+60x3)/(1+3）=70。这就是根据权重的不同进行的平均数的计算，所以又叫加权平均数

权重变量的不同将极大地影响图形内容以及观察结论, 有两种可以用于表现权重的可调图形属性. 首先, 对于线和点这类简单点几何对象, 我们可以根据点点数量调整图形属性size来改变点的大小:

qplot(percwhite, percbelowpoverty, data=midwest, size=poptoal/1e6)+scale_size_area("Pupulation\n(millinons)", breaks=c(0.5, 1, 2, 4))

qplot(percwhite, percbelowpoverty, data=midwest, size=area)+scale_size_area()

对于更复杂的, 涉及到统计变换的情况, 我们通过修改weight图形属性来变现权重:

lm_samoth <- geom_samooth(method=lm, size=1)

qplot(percwhite, percbelowpoverty, data=midwest) + lm_smooth

qplot(percwhite, percbelowpovety, data=midwest, weight=popdensity, size=popdensity) + lm_smooth

qplot(percbelowpoverty, data=midwest, weight=poptotal, binwidth=1) + ylab("population")

标度(scale)控制着数据到图形属性的映射， 将我们的数据转化视觉上可感知的东西: 大小, 颜色, 位置或形状. 同时也提供了读图时所使用的工具: 坐标轴和图例, 更准确地说, 每一个标度都是从数据空间的某个区域(标度的定义域)到图形属性空间的某个区域(标度的值域)的一个函数.

标度的种类依赖于变量的类型: 标度可为连续型(变量为数值时)或离散型(变量为因子, 逻辑值, 字符时)

标度构建器(scale constructor)都拥有一套通用的命名方案, 它们以scale_开头, 接下来时图形属性的名称(colour_, shFtape_, x_), 最后以标度的名称结尾(gradient, hue, manual)

p + salce_colour_hue()

p + scale_colour_brewer(palette="Set1")

p + scale_colour_hue("What does\n it eat?")

p + scale_colour_gradient(low="green", high="red",breaks=c(0,1))

breaks=c("herbi","carni","omni",NA),labels=c("plants","meat","both","Don't know"))

标量通用参数

name: 设置坐标轴或图例上出现的标签，使用三个辅助函数xlab(), ylab(), labs()减少部分键入

p + scale_x_continuous("City mpg")

p + xlab("City mpg")

p + ylab("Highway mpg")

p + labs(x="City mpg", y="Highway", colour="Displacement")

p + xlab(expression(frac(miles, gallon)))

limits: 固定标度的定义域: 连续型标度接受一个长度为2的数值型向量; 离散型标度接受一个字符型向量, 一旦设定了limits, 数据将不再进行任何训练, 任何不在此标度定义域内的值将被丢弃

breaks和labels: breaks控制着显示在坐标轴或图例上的值, 即, 坐标轴上应该显示哪些刻度线的值, 或一个连续型标度在一个图例中将被如何分段. labels指定了应在断点处显示的标签. 若设置了labels, 则必须同时指定breaks, 只有这样, 这两个参数才能被正确匹配. breaks影响显示在坐标轴和图例上的元素, 而limits影响显示在图形上的元素.

p + scale_x_continuous(breaks=c(5.5, 6.5), labels=c("tag_a", "tag_b"))

位置标度: xlim(10, 20), 一个从10到20的连续型标度; ylim(20, 10), 一个从20到10的反转后连续型标度(图像也会发生发转); xlim("a","b","c"), 一个离散型标度; xlim(as.Date(c("2008-05-01", "2008-08-01"))), 一个从2008年5月1日到8月1日的日期型标度

连续型: scale_x_continuous和scale_y_continuous, 它们均将数据映射到x轴和y轴, 最有趣的变式是通过变换来生成的, 每个连续型标度均可接受一个trans参数, 允许指定若干种线性或非线性的变换

scale_x_log10()和scale_x_continous(trans="log10")等价

log10(x)和scale_x_log10(), 在绘图区域生成完全相同的结果, 但坐标轴和刻度标签却是不同的, log10(x)为转换后值, scale_x_log10()为转换前值

日期和时间: as.Date(), as.POSIXct()

三个参数可选, major, minor, format. major="2 weeks"将在每隔2周的位置放置一个主刻度. format指定了刻度标签的格式, 14/10/1979的形式显示，需使用字符串"%d/%m/%y"

自动选择颜色离散型标度:

point + scale_colour_brewer(palette="Set1")

point + scale_colour_brewer(palette="Set2")

point + scale_colour_brewer(palette="Pastel1")

手动离散型标度:

scale_shape_manual(); scale_linetype_manual(); scale_colour_manual()

scale_colour_manual(values=c("red","orange","yellow","green","blue"))自动填充

colours <- c(carni="red", "NA"="orange", insecti="yellow",herbi="green",omni="blue"); scale_colour_manual(values=colours)指定填充

ggplot(huron, aes(year)) + **geom_line(aes(y=level-5, colour="below")) + geom_line(aes(y=level+5, colour="above")) + scale_colour_manual("Direction", values=c("below"="blue", "above"="red"))** 这里对应aes(color=group)要指定！！！

---

ggplot2提供两种分面类型: 网格型(facet_grid)和封装型(facet_wrap). 网格分面生成一个2维的面板网格, 面板的行与列通过变量来定义; 封装分面则先生成一个1维的面板条块, 然后再封装到2维中

qplot()中也可以选择分面系统. 2维分面(x~y)使用facet_grid, 1维分面(~x)使用facet_wrap

mpg2 <- subset(mpg, cyl != 5 && drv %in% c("4","f"))

" . ~ a": 一行多列

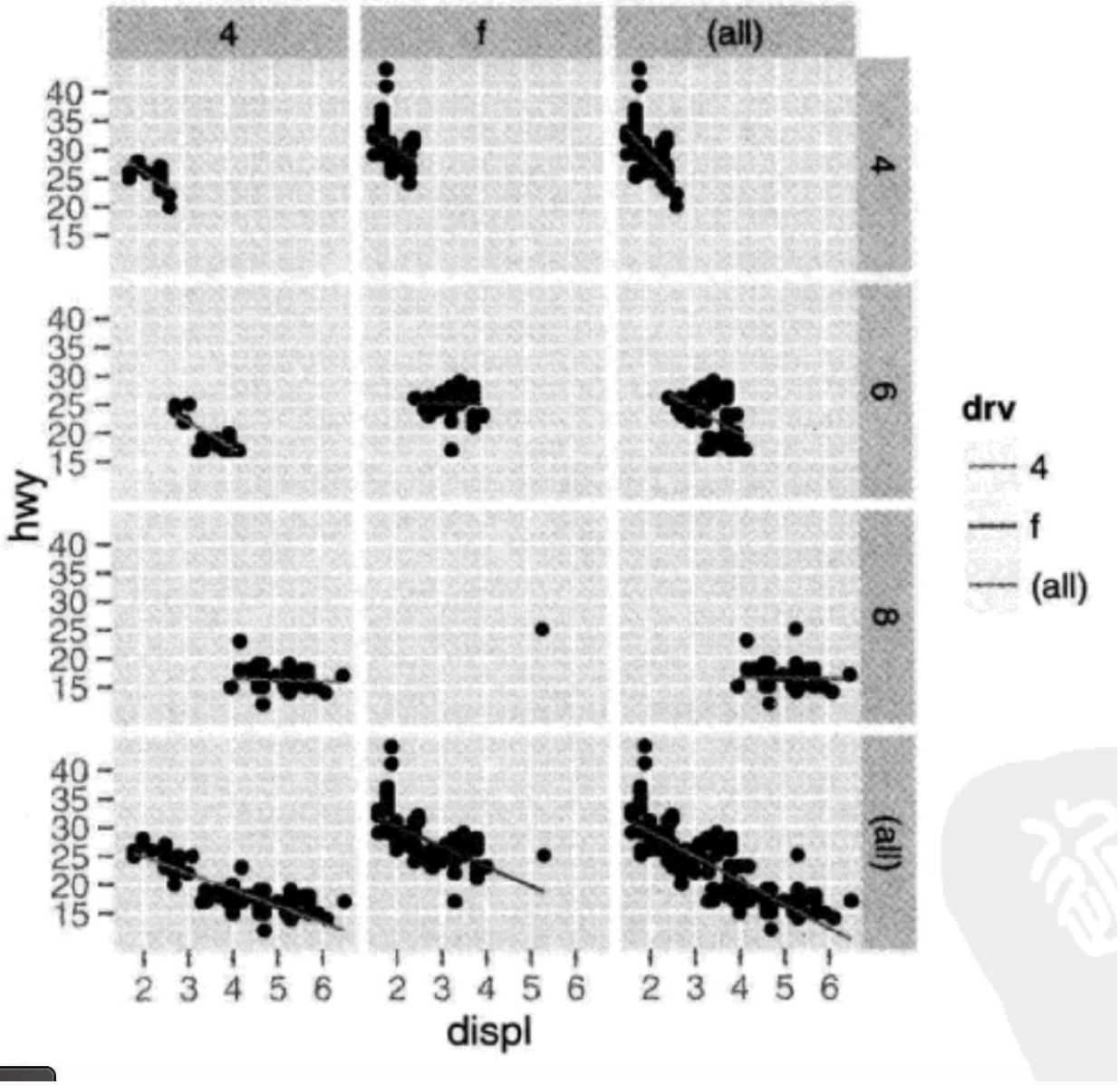
"b ~ .": 一列多行

"a ~ b": 多行多列, 通常将因子水平数目最大的变量按列排放, 这样充分利用屏幕的宽高比

". ~ a+b ; a+b ~ .": 多个变量的多个水平在行或着列上(或者同时), 不太实用

边际图就好比创建一个列联表, 列联表可展示每个单元格的值以及边际和(一个行或列的总和), 可以使用参数margins来绘制边际图: 设定margins=TRUE可展示所有边际图, margins=c("sex","age")列出要展示的边际图的变量名称, 另外使用grand_row或grand_col来分别生成所有行或列的边际图

facet_grid(cyl ~ drv, margins=T)



facet_wrap, 先生成一个长的面板条块(由任意数目的变量生成), 然后将它封装在2维中: 分面变量的设置形式为~a+b+c, facet_wrap默认把图形面板尽可能摆成方形, 且更偏好于宽扁型的矩形. 可以通过设置ncol,nrow来更新默认设置.

facet_wrap(~decade, ncol=6)

对于以上两种分面, 可以调整参数scales来控制面板的位置标度是相同(固定)还是允许变化(自由):

scales="fixed": x和y的标度在所有面板中都相同

scales="free": x和y的标度在每个面板都可以变化

scales="free_x": x的标度可变, y的尺度固定

scales="free_y": y的标度可变，x的尺度固定

facet_wrap(~cyl, scales="free")

facet_grid还有一个额外的参数space, 值可为"free"或"fixed", 当space设定为free时, 每列(行)的宽度(高度)与该列(行)的标度范围成比例

reorder()可使模型(model)和生厂商(manufacturer)按照城市有油耗(cry)重新排序, 第一个变量为分类变量, 第二个变量为数值变量

或者通过factor设定levels来在坐标轴排序

mpg3 <- within(mpg2, { model <- reorder(model, cty)

manufacturer <- reorder(manufacturer, -cty)})

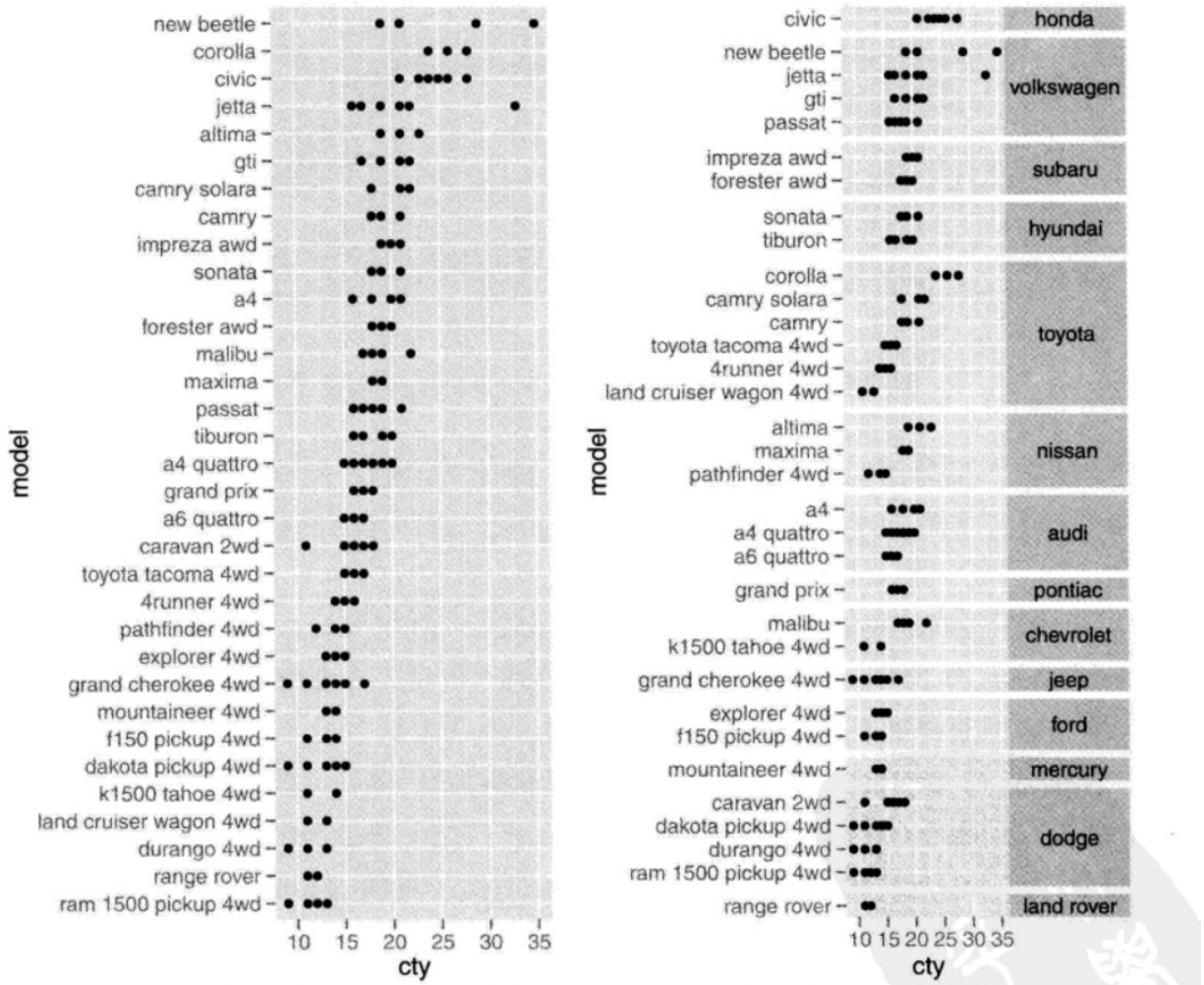


图 7.6 每种小汽车的城市油耗英里数的点图。(左) 车类型按 mpg 均值排序，(右) 使用 `scales = "free_y"` 和 `space = "free"`，按生产商进行分面。`strip.text.y` 主题设置用来旋转分面标签。

对连续变量进行分面,首先需要将其变换成离散型: 将数据分为n个长度相同的部分, 用cut_interval(x, n=10)控制划分数目, 或用cut_interval(x, length=1)控制每个部分的长度

mpg2$disp_ww <- cut_inteval(mpg2$displ, length=1)

qplot(cty, hwy, data=mpg2) + labs(x=NULL, y=NULL) + facet_wrap(~disp_ww, nrow=1)

内置主题有两种, 默认的theme_gray()使用淡灰色背景和白色网格线; 另一个固定主题theme_bw()为传统的白色背景和深灰色的网格线: 两个主题都是有唯一的参数base_size来控制基础字体的大小, 基础字体大小指的是轴标题的大小, 图形标题比它大20%, 轴须标签比它小20%。

全局性设置: theme_set(theme_gray())或theme_set(theme_bw()), theme_set()返回先前的主题, 可储存以备后用

局部性设置: 只改变单个图形的主题, qplot(...)+theme_gray(), 局部设置将会覆盖默认的全局性设置

theme_set(theme_bw())

qplot(rating, data=movies, binwidth=1)+theme_set(theme_bw())

主题由控制图形外观的多个元素组成, 有三个元素含有x和y的设置: axis.text, axis.title和strip.text. 通过对水平和竖直方向元素的不同设置, 我们可控制不同方向的文本外观, 这些控制元素外观的函数被称为元素函数

element_text()绘制标签和标题, 可控制字体的family, face, colour, size, hjust, vjust, angle, lineheight

hgramt <- hgram + labs(title="This is a histogram")

hgramt + theme(plot.title=element_text(size=20, colour="red", hjust=0, face="bold", angle=180)) 改变图形标题外观

element_line()绘制线条或线段, 该元素函数可控制colour, size, linetype

hgram + theme(panel.grid.major=element_line(colour="red", size=2, linetype="dotted"))

hgram + theme(axis.line=element_line(colour="red", size=0.5, linetype="dashed")) 改变图形中线条或线段的外观

eg.
ggplot(mpg,aes(manufacturer,displ))+geom_point(position="jitter")+geom_boxplot(alpha=1/100)+theme(axis.text.x=element_text(angle=90, hjust=1)) 旋转x坐标标签值

element_rect()绘制主要背景使用的矩形, 可以控制填充颜色(fill)和边界的colour, size, linetype

hgram + theme(plot.background=element_rect())

hgram + theme(plot.background=element_rect(colour="red", size=2, fill="grey80", linetype="dotted")) 改变图形和面板背景的外观

element_blank()表示空主题, 即对元素部分配相应的绘图空间, 该函数可删去我们不感兴趣的绘图元素, 使用之前的colour=NA, fill=NA让某些元素不可见, 可达到相同的效果, 但仍占绘图空间

hgramt + theme(panel.grid.minor=element_blank(), panel.grid.major=element_blank(), panle.background=element_blank(), axis.title.x=element_blank(), axis.title.y=element_blank(), axis.line=element_blank())

hgramt + theme(element_blank())

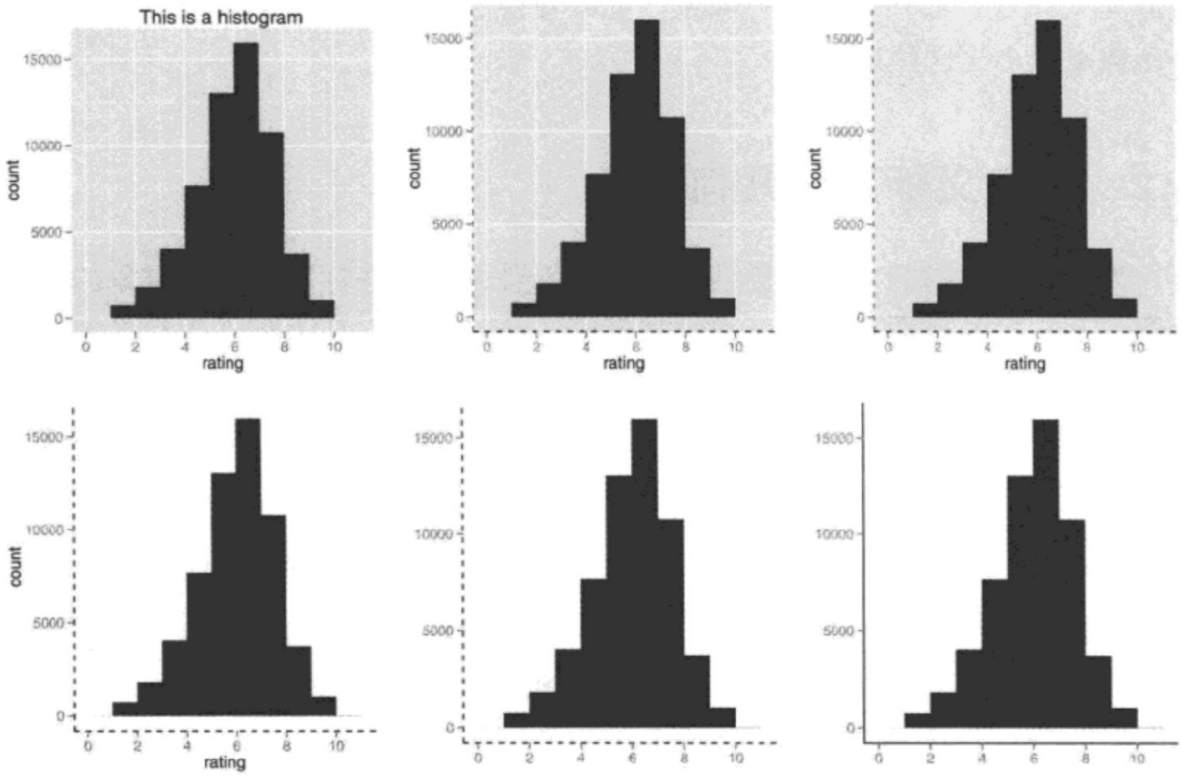


图 8.5 使用 element_blank() 逐步从图中删除非数据元素。

使用theme_get()可得到当前主题的设置, theme()可在一副图中对某些元素进行局部性修改, theme_update()可为后面图形的绘制进行全局性地修改

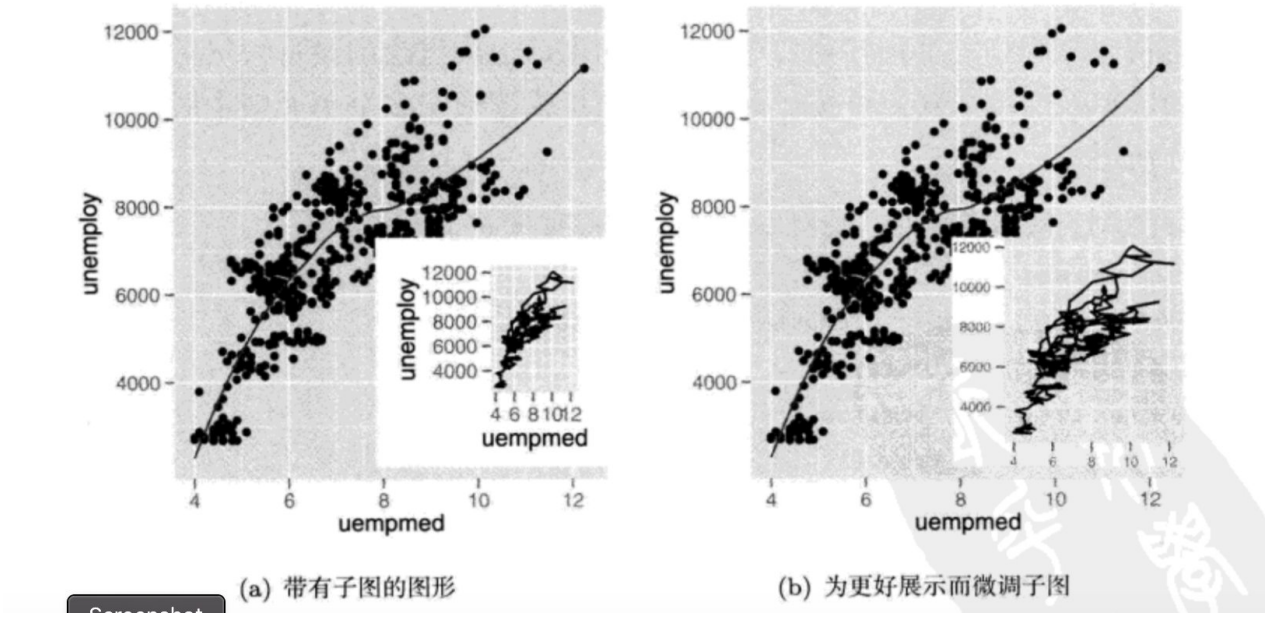
一页多图, 其关键概念是视图窗口(viewport): 显示设备的一个矩形子区域, 默认的视图窗口是占据了整个绘图区域, 通过设置视图窗口, 可任意安排多幅图形的位置

viewport()函数可创建视图窗口,参数x,y表示子图位置, width,height表示窗口大小, 同时可以采用unit(2, "cm")或unit(1, "inch")这样的绝对单位控制窗口大小

```
library(grid)

b <- qplot(uempmed, unemploy, data=economics) + geom_smooth(se=F)

c <- qplot(uempmed, unemploy, data=economics, geom="path")

subvp <- viewport(width=0.4, height=0.4, x=0.75, y=0.35)

vp3 <- viewport(width=unit(2, "cm"), height=unit(3, "cm"))
```

使用x,y控制视图窗口的中心位置,若想调整图形位置, 可通过just参数来控制将图形放置在哪个边角, 若just为2个标量, 第一个为水平位置, 第二个为垂直位置: left, right, center, top, bottom

```
b; print(c, vp=subvp); dev.off()
```



注意需要使用pdf()或png()将图形存储到磁盘中, 因为ggsave()只能存储一副图

矩形网格, grid.layout(), 它设置了一个任意高和宽的视图网格, 仍需一个一个创建视图窗口, 但是不用设置视图窗口的位置和大小, 只需设置布局(layout)的行数和列数即可

library(grid)

grid.newpage()

pushViewport(viewport(layout=grid.layout(2,2)))

vplayout <- function(x,y){

viewprot(layout.pos.row=x, layout.pos.col=y)}

layout.pos.row: A numeric vector giving the rows occupied by this viewport in its parent's layout

layout.pos.col: A numeric vector giving the columns occupied by this viewport in its parent's layout

print(a, vp=vplayout(1,1:2))

print(b, vp=vplayout(2,1))

print(c, vp=vplayout(2,2))

dev.off()

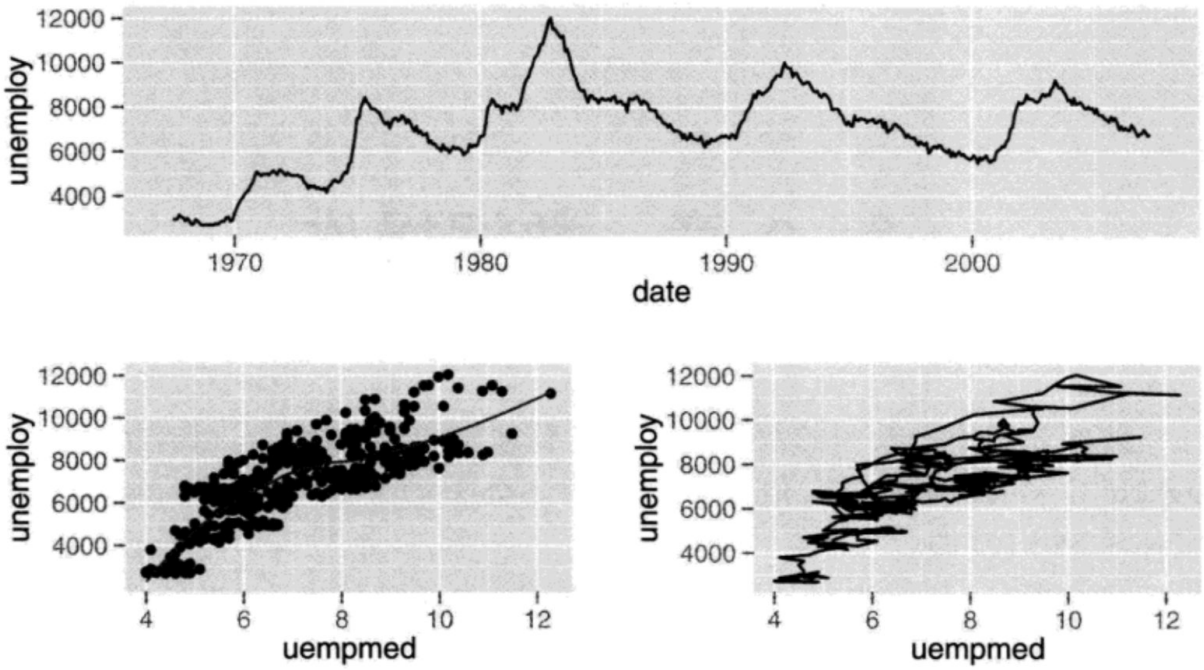
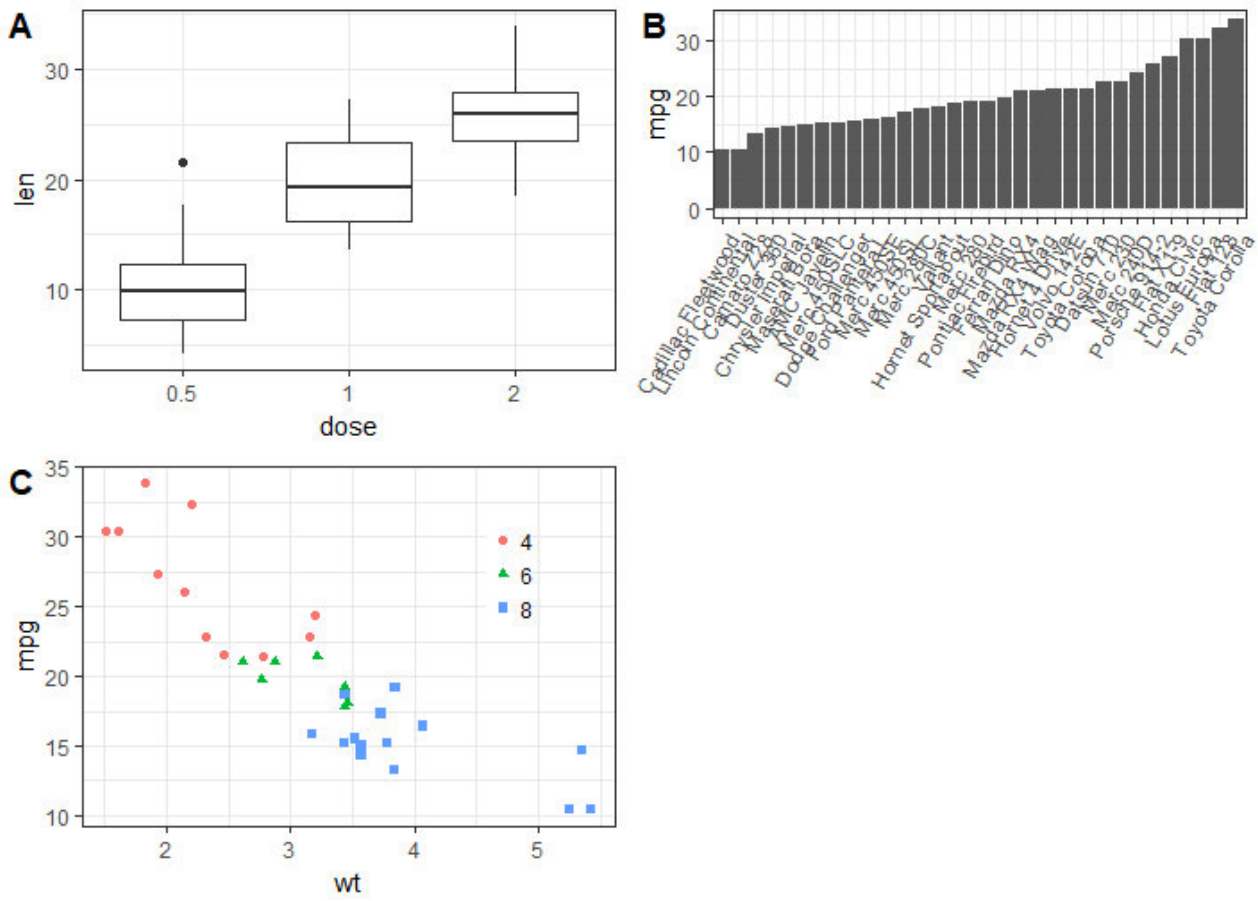


图 8.10 grid.layout() 将三幅图形分置在一页上。

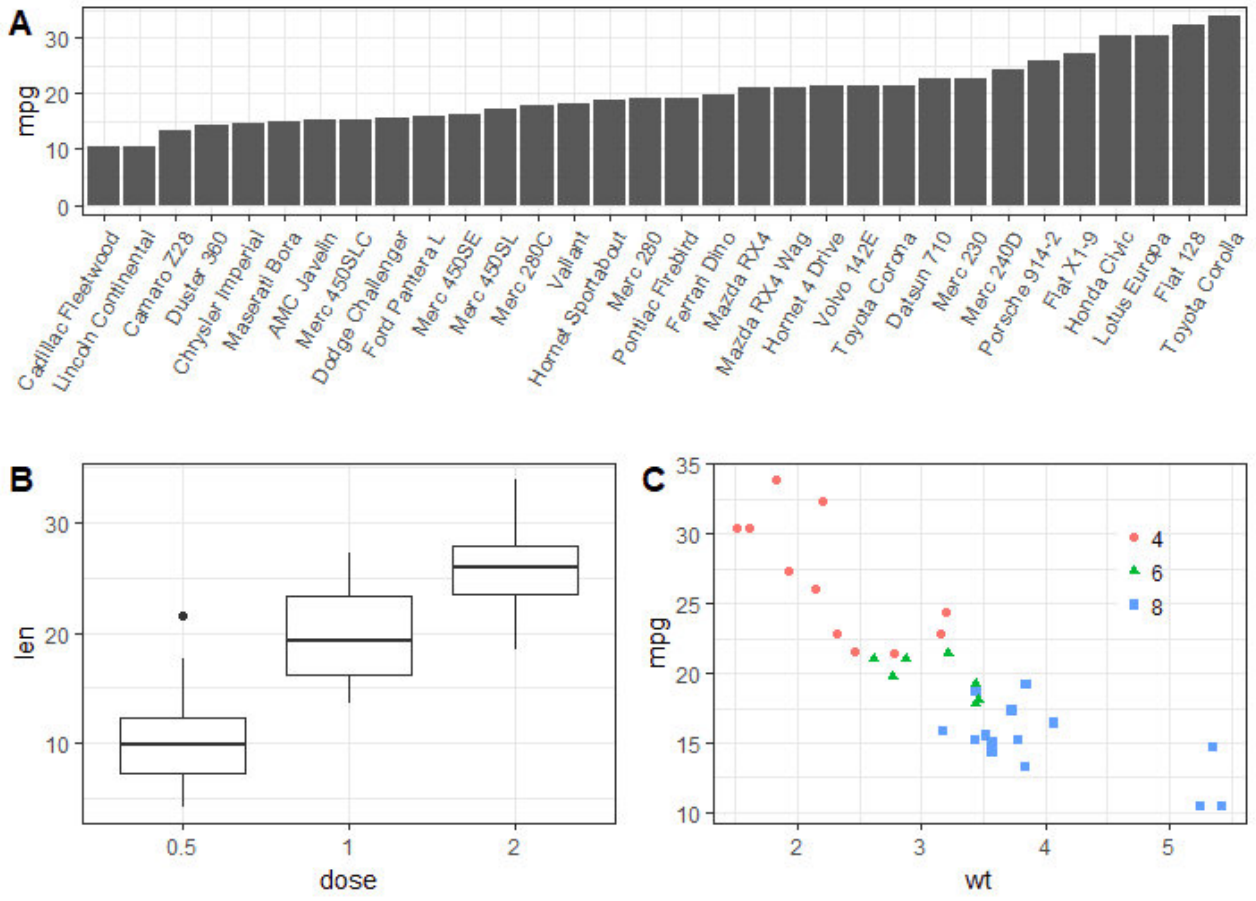
默认grid.layout()中, 每个单元格的大小都相同, 可设置widths和heights参数使它们具有不同的大小

使用package, ggpubr实现多图合并一张

```
library(ggpubr)
data("ToothGrowth")
data("mtcars")
P1 <- ggplot(ToothGrowth,aes(dose,len,group=factor(dose))+geom_boxplot())
p2 <- ggplot(mtcars,aes(x=reorder(name,mpg),y=mpg))+geom_bar(stat="identity")
p3 <- ggplot(mtcars, aes(wt,mpg,color=factor(cyl),shape=factor(cyl)))+geom_point()
ggarange(p1,p2,p3,ncol=2,nrow=2,labels=c("A","B","C"))
```



ggarrange(p2, ggarrange(p1,p3,ncol=2,labels=c("B","C")),now=2,lables="A")



ddply(.data, .variable, .fun, ...)

.data是用来做图的数据

.variable是对数据取子集的分组变量, 形式是.(var1,var2), 为了与图形保持一致, 该变量必须包含所有在画图过程中用到的分组变量和分面变量

.fun是要在各个子集上运行的统计汇总函数, 这个函数可以返回向量也可以返回数据框

subset()用来对数据取子集的函数, 选择数据中前(或后)n个(或x%的)观测值, 或是在某个阈值之上或下的观测值

ddply(diamonds, .(color), subset, carat == min(carat))

ddply(diamonds, .(color), subset, order(carat)<=2)

ddply(diamonds, .(color), subset, carat > quantile(carat,0.99))

ddply(daimonds, .(color), subset, price > mean(price))

transform()是用来进行数据变换的函数, 与deploy()一起可以计算分组统计量, 例如各组的标准差, 并且加到原来数据上去

ddply(diamonds, .(color), transform, price = scale(price)) ddply(diamonds, .(color), transform, price = price - mean(pirce))

colwise()用来向量化一个普通函数, 也就是说colwise()能把原来只接受向量输入到函数变成可以接受数据框输入到函数

nmissing <- function(x)sum(is.na(x))

colwise(nmissing)(msleep)

numcolwise()是colwise()的一个特殊版本, 功能类似, 但numcolwise()只对数值类型的列操作:
numcolwise(median)对每个数值类型的列计算中位数, numcolwise(quantile)对每个数值类型的列计算分位数

numcolwise(medain)(msleep, na.rm=T)

numcolwise(quantile,probs=c(0.25,0.75))(msleep, na.rm=T)

ddply(msleep, .(vore), numcolwise(mean), na.rm=T)

ddply(msleep, .(vore), numcolwise(mean, na.rm=T))

可手动编写函数, 只要它能接收和输出数据框即可

stat_smooth: 'loess()' 用于小于1000个观察值, 否则就采用'mgcv::gam()', 公式为'formula = y ~ s(x, bs="cs")'

library(mgcv)

dense <- subset(diamonds, carat <2)

smooth <- function(df){ mod <- gam(price ~ s(carat, bs="cs"), data=df)

grid <- data.frame(carat=seq(0.2,2,length=50))

pred <- predict(mod, grid, se=T) grid$price <- pred$fit

```
grid$se <- pred$se.fit

grid

}

smoothes <- ddply(dense, .(color),smooth)

qplot(carat, price, data=smoothes, colour=color, geom="line")


mod <- gam(price ~ s(carat, bs="cs") + color, data=dense)

grid <- with(diamonds, expand.grid(

carat = seq(0.2,2,length=50),

color = levels(color)

))

grid$pred <- predict(mod, grid)

qplot(carat, pred, data=grid, colour=color, geom="line")
```

expand.grid: Create a data frame from all combinations of factor vairables

data.frame(carat=seq(0.2,2,length=50),color=levels(diamonds$color))Error in data.frame(carat = seq(0.2, 2, length = 50), color = levels(diamonds$color)) : arguments imply differing number of rows: 50, 7

ggplot2进行数据分组时必须根据行, 而不能根据列: melt()函数有三个参数

data: 待变形的原数据

id.vars: 依旧放在列上, 位置保持不变的变量, id.vars通常是离散的, 并且是预先给定的

measure.vars: 需要被放进同一列的变量, 不同的变量放在同一列后, 根据原变量名来分组, 这些变量就可以同时画在一张图里

多重时间序列

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

geom_line(aes(y=unemploy,colour="unemploy"))+

geom_line(aes(y=uempmed,colour="unemploy"))+

scale_colour_hue("variable")

require(reshape2)

emp <- melt(economics, id="date", measure=c("unemploy", "uempmed"))

qplot(date, value, data=emp, geom="line", colour=variable)

ggplot(emp, aes(date, value))+geom_line(aes(color=variable))+
```

scale_colour_manual("variable",values=c(unemploy="green",uempmed="blue"))

两变量取值差异太大:

Range01 <- function(x){

rng <- range(x, na.rm = t)

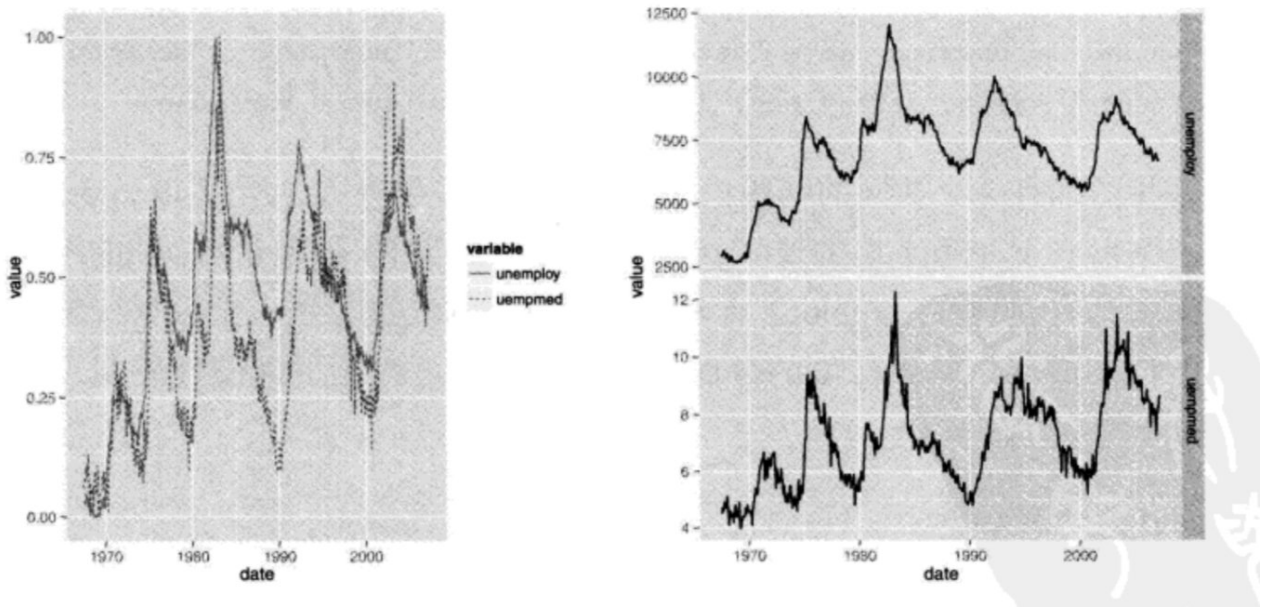
(x - rng[1])/diff(rng)

}

emp2 <- ddply(emp, .(variable), transform, value=range01(value))

qplot(date, value, data=emp2, geom="line", colour=variable, linetype=variable)

qplot(date, value, data=emp, geom="line", color=variable) + facet_grid(variable~., scales="free_y")



平行坐标图: 就是以varaible变量为x轴表示变量名, 以value为y轴表示变量取值, 此外, 还需要一个分组变量来把各个观测分组(所以每个观测分别对应一条线)

popular <- subset(movies, vote > 1e4)

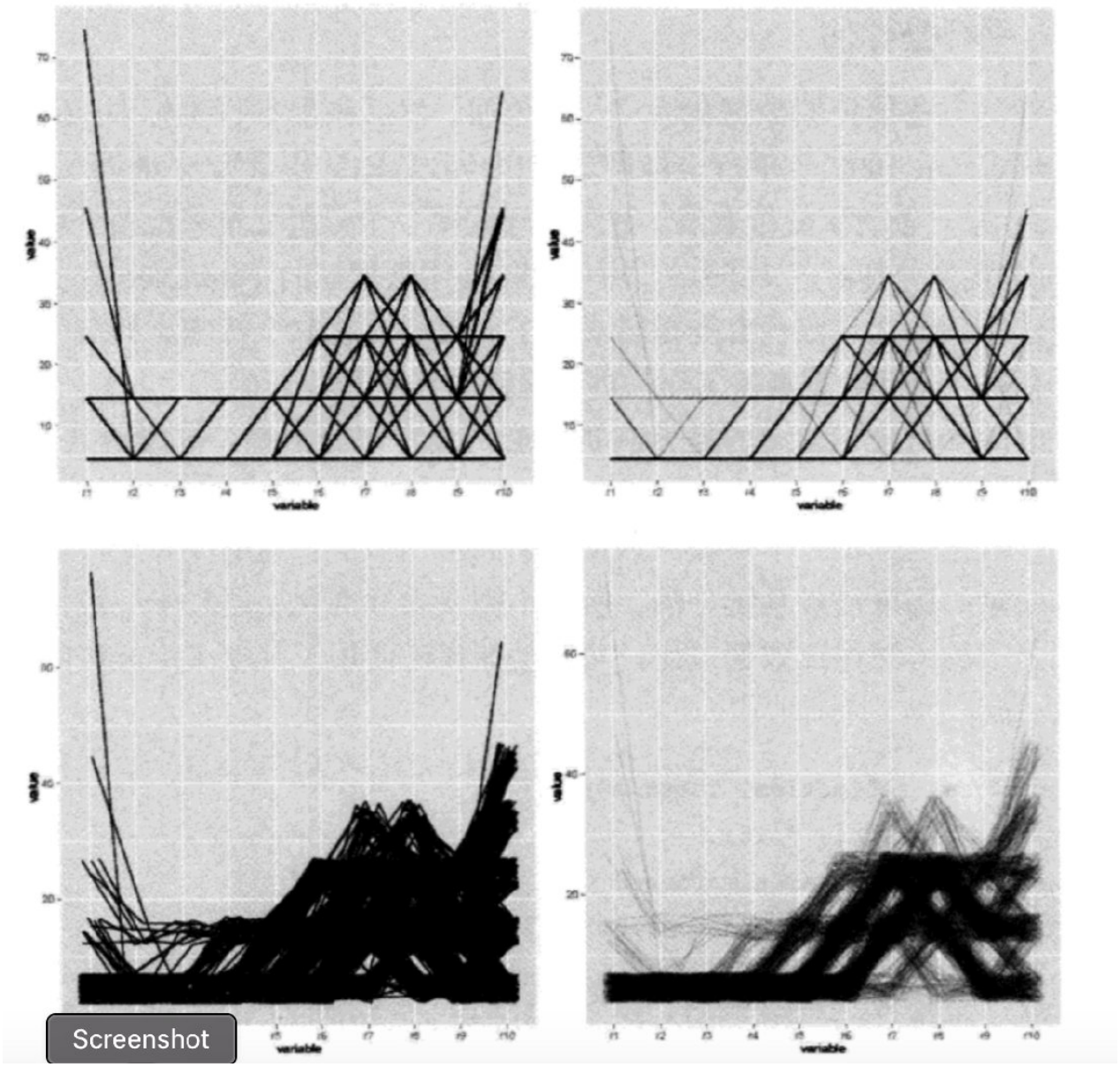
ratings <- popular[, 7:16]

ratings$.row <- rownames(ratings)

molten <- melt(ratings, id=".row")

此时以variable为x轴, 以value为y轴, 以.row分组, 画出折线, 就得到了平行坐标图

ggplot(molten, aes(variable, value, group=.row))+geom_line(colour="black", alpha=1/20, position="jitter")



使用kmeans聚类, 将所有电影分为6类, 平均分最低的记做第一类, 平均分最高的记做第六类

cl <- kmeans(ratings[1:10], 6)

ratings$cluster <- reorder(factor(cl$cluster), popular$rating)

levels(ratings$cluster) <- seq_along(levels(ratings$cluster))

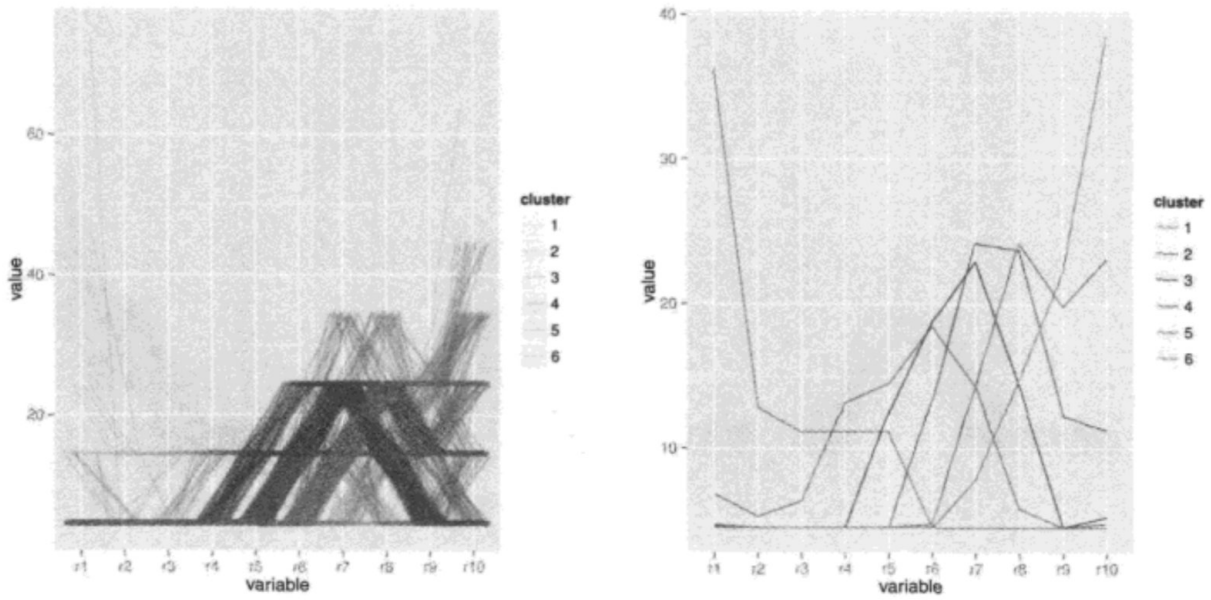
molten <- melt(ratings, id=c(".row", "cluster"))

kmeans: Perform k-means clustering on a data matrix.

kmeans(x, centers, iter.max = 10, nstart = 1, algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"), trace=FALSE)

pcp_cl <- ggplot(molten, aes(variable, value, group=.row, colour=cluster))

pcp_cl + geom_line(position="jitter", alpha=1/5)



pcp_cl + geom_line(position="jitter", colour="black", alpha=1/5) + facet_wrap(~cluster)

Miscellaneous

- geom_bar

```
p <- ggplot(data=df, aes(x=dose, y=len)) +geom_bar(stat="identity")
```

`p + coord_flip()`, 颠倒

- 修改图示

labs(title="Title") + xlab("x") + ylab("y") ###添加图示

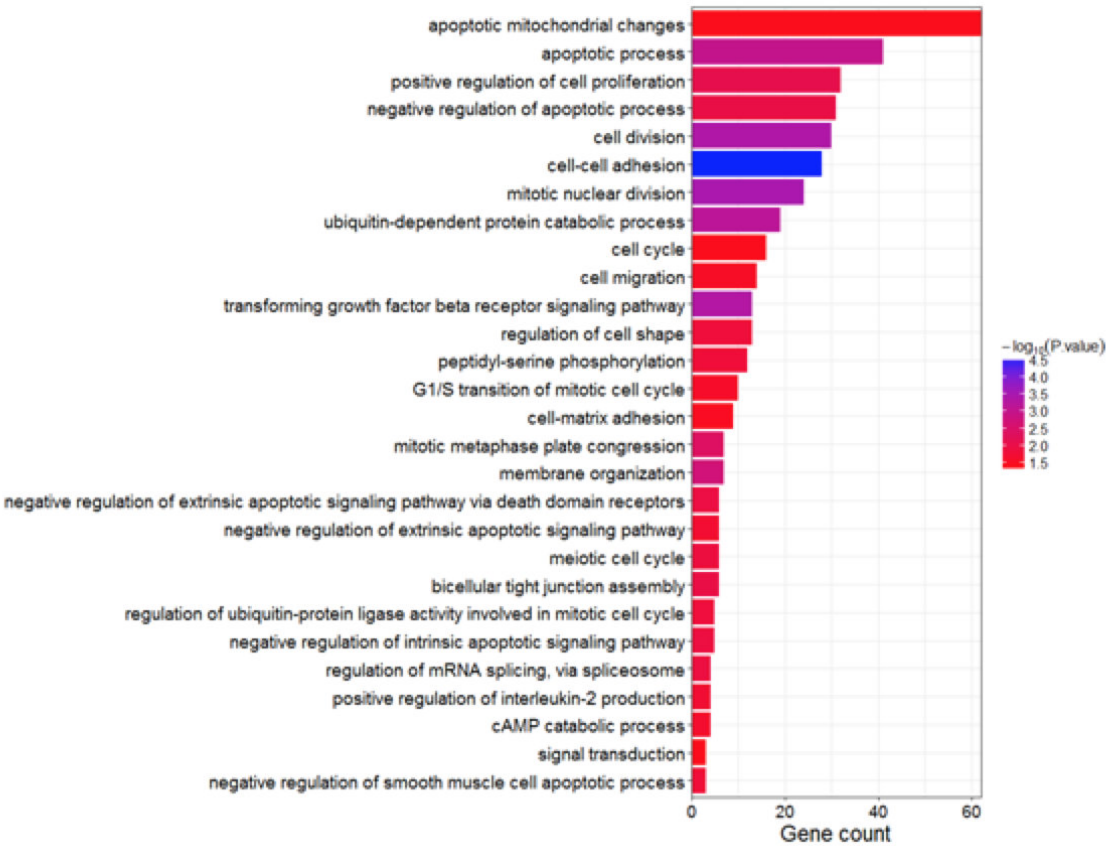
labs(title="Title", x= "X", y="Y") ###添加图示

theme(legend.text=element_text(size=6))###对应修改图示和图例

theme(plot.title=element_text(hjust=0.5))###对应修改title位置

[富集条形图][https://www.jianshu.com/p/462423702851]：

```
rm(list=ls())
library(ggplot2)
library(Cairo)

setwd("/home/ ")
GO_BP <- read.table("./enh_statistics/A549_GO_BP_spe.tsv",header = T,sep="\
t")

png_path="./figure/GO_BP.png"
CairoPNG(png_path, width = 12, height = 7, units='in', dpi=600)

ggplot(data=GO_BP)+
  geom_bar(aes(x=reorder(Term,Count),y=Count, fill=-log10(PValue)), stat='i
dentity') +
  coord_flip() +
  scale_fill_gradient(expression(-log["10"](P.value)),low="red", high = "bl
ue") +
  xlab("") +
  ylab("Gene count") +
  scale_y_continuous(expand=c(0, 0))+
  theme(
    axis.text.x=element_text(color="black",size=rel(1.5)),
    axis.text.y=element_text(color="black", size=rel(1.6)),
    axis.title.x = element_text(color="black", size=rel(1.6)),
    legend.text=element_text(color="black",size=rel(1.0)),
    legend.title = element_text(color="black",size=rel(1.1))
    # legend.position=c(0,1),legend.justification=c(-1,0)
    # legend.position="top",
    )
dev.off()
```


apoptotic mitochondrial changes
apoptotic process
positive regulation of cell proliferation
negative regulation of apoptotic process
cell division
cell-cell adhesion
mitotic nuclear division
ubiquitin-dependent protein catabolic process
cell cycle
cell migration
transforming growth factor beta receptor signaling pathway
regulation of cell shape
peptidyl-serine phosphorylation
G1/S transition of mitotic cell cycle
cell-matrix adhesion
mitotic metaphase plate congression
membrane organization
negative regulation of extrinsic apoptotic signaling pathway via death domain receptors
negative regulation of extrinsic apoptotic signaling pathway
meiotic cell cycle
bicellular tight junction assembly
regulation of ubiquitin-protein ligase activity involved in mitotic cell cycle
negative regulation of intrinsic apoptotic signaling pathway
regulation of mRNA splicing, via spliceosome
positive regulation of interleukin-2 production
cAMP catabolic process
signal transduction
negative regulation of smooth muscle cell apoptotic process
0
20
40
60
Gene count
−log10(P value)
4.5
4.0
3.5
3.0
2.5
2.0
1.5

气泡图：

```
rm(list=ls())
library(Cairo)
library(stringr)
setwd("/home/")

pathway = read.table("./enh_statistics/A549_KEGG.tsv",header=T,sep="\t")
pathway$Term<-str_split_fixed(pathway$Term,":",2)[,2]

png_path="./figure/KEGG.png"
CairoPNG(png_path, width = 5.9, height = 3, units='in', dpi=600)

ggplot(pathway,aes(x=Fold.Enrichment,y=Term)) +
  geom_point(aes(size=Count,color=-1*log10(PValue)))+
  scale_colour_gradient(low="green",high="red")+
  labs(
       color=expression(-log[10](P.value)),
       size="Gene number",
       x="Fold enrichment"
       # y="Pathway name",
       # title="Pathway enrichment")
      )+
  theme_bw()+
  theme(
    axis.text.y = element_text(size = rel(1.3)),
    axis.title.x = element_text(size=rel(1.3)),
    axis.title.y = element_blank()
  )
dev.off()
```