ggplot2数据可视化::速查表

基础

ggplot2 基于图形语法，使用相同的组件（数据集、坐标系统和表示数据点的几何对象）来构建图片。

为了获取显示值，数据中的变量映射到图形的视觉属性，如大小、颜色以及x和y位置。

完成以下模板来构建图形

```
ggplot (data = <DATA> ) +                          必要
<GEOM_FUNCTION> (mapping = aes( <MAPPINGS> ),
 stat = <STAT> , position = <POSITION> ) +         非必要，有默认值
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>
```

ggplot(data = mpg, **aes(**x = cty, y = hwy**))** 通过添加图层来完成图形，每层添加一个geom函数。

映射 数据 图形

qplot(x = cty, y = hwy, data = mpg, geom = “point"**)** 用给定的数据、几何对象和映射创建完整的图片。 绘图函数提供许多有用的默认设置。

last_plot() 返回上一个图片

ggsave("plot.png", width = 5, height = 5)将最后一个图片保存至工作目录中名为“plot.png”的5'x 5'文件。 文件类型与文件扩展名相匹配。

几何对象

使用geom函数表示数据点，使用geom的属性表示变量。 每个函数绘制一个图层。

基本图像

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank()
(Useful for expanding limits)

b + geom_curve(aes(yend = lat + 1, xend=long+1,curvature=z)**)** - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend="butt", linejoin="round", linemitre=1**)**
x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(group = group)**)**
x, y, alpha, color, fill, group, linetype, size

b + geom_rect(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1)**)** - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900)**)** - x, ymax, ymin, alpha, color, fill, group, linetype, size

分段线

常用参数: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept=0, slope=1)**)**
b + geom_hline(aes(yintercept = lat)**)**
b + geom_vline(aes(xintercept = long)**)**

b + geom_segment(aes(yend=lat+1, xend=long+1)**)**
b + geom_spoke(aes(angle = 1:1155, radius = 1)**)**

单一变量 连续

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian"**)**
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly() x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5**)** x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy)**)** x, y, alpha, color, fill, linetype, size, weight

离散

d <- ggplot(mpg, aes(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

双变量

连续x、连续y

e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE**)** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_jitter(height = 2, width = 2**)**
x, y, alpha, color, fill, shape, size

e + geom_point(), x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile(), x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl"**)**, x, y, alpha, color, linetype, size

e + geom_smooth(method = lm**)**, x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE**)**, x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

离散x，连续y

f <- ggplot(mpg, aes(class, hwy))

f + geom_col(), x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot(), x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center"**)**, x, y, alpha, color, fill, group

f + geom_violin(scale = "area"**)**, x, y, alpha, color, fill, group, linetype, size, weight

离散x，离散y

g <- ggplot(diamonds, aes(cut, color))

g + geom_count(), x, y, alpha, color, fill, shape, size, stroke

连续二元分布

h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500)**)**
x, y, alpha, color, fill, linetype, size, weight

h + geom_density2d()
x, y, alpha, colour, group, linetype, size

h + geom_hex()
x, y, alpha, colour, fill, size

连续函数

i <- ggplot(economics, aes(date, unemploy))

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

误差的呈现方式

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

j + geom_crossbar(fatten = 2**)**
x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar(), x, ymax, ymin, alpha, color, group, linetype, size, width (also **geom_errorbarh()**)

j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange()
x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

地图

data <- data.frame(murder = USArrests$Murder,
state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

k + geom_map(aes(map_id = state), map = map**)**
+ expand_limits(x = map$long, y = map$lat**)**,
map_id, alpha, color, fill, linetype, size

三变量

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))l <- ggplot(seals, aes(long, lat))

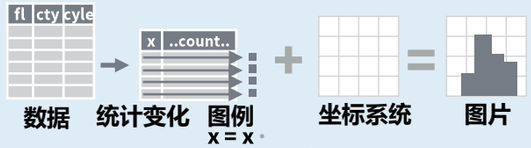
l + geom_contour(aes(z = z))
x, y, z, alpha, colour, group, linetype, size, weight

l + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE)
x, y, alpha, fill

l + geom_tile(aes(fill = z)), x, y, alpha, color, fill, linetype, size, width

统计变换 另一种构建图层的方法

统计变化构建新变量来绘图（例如，count，prop）。



通过更改geom函数的默认统计信息，
geom_bar(stat="count") 或者使用统计变化功能来绘图**stat_count(geom="bar")**，其调用默认图片来创建一个图层（相当于geom函数）。
使用 **..name..** 语法将统计变化映射到坐标。

使用的绘图函数 | 统计函数 | 映射

i + stat_density2d(aes(fill = ..level..), geom = "polygon")

由统计变换创建的变量

c + stat_bin(binwidth = 1, origin = 10)
x, y | ..count.., ..ncount.., ..density.., ..ndensity..

c + stat_count(width = 1) **x, y,** | ..count.., ..prop..

c + stat_density(adjust = 1, kernel = "gaussian")
x, y, | ..count.., ..density.., ..scaled..

e + stat_bin_2d(bins = 30, drop = T)
x, y, fill | ..count.., ..density..

e + stat_bin_hex(bins=30) **x, y, fill** | ..count.., ..density..

e + stat_density_2d(contour = TRUE, n = 100)
x, y, color, size | ..level..

e + stat_ellipse(level = 0.95, segments = 51, type = "t")

l + stat_contour(aes(z = z)) **x, y, z, order** | ..level..

l + stat_summary_hex(aes(z = z), bins = 30, fun = max)
x, y, z, fill | ..value..

l + stat_summary_2d(aes(z = z), bins = 30, fun = mean)
x, y, z, fill | ..value..

f + stat_boxplot(coef = 1.5) **x, y** | ..lower.., ..middle.., ..upper.., ..width.. , ..ymin.., ..ymax..

f + stat_ydensity(kernel = "gaussian", scale = "area") **x, y** | ..density.., ..scaled.., ..count.., ..n.., ..violinwidth.., ..width..

e + stat_ecdf(n = 40) **x, y** | ..x.., ..y..

e + stat_quantile(quantiles = c(0.1, 0.9), formula = y ~ log(x), method = "rq") **x, y** | ..quantile..

e + stat_smooth(method = "lm", formula = y ~ x, se=T, level=0.95) **x, y** | ..se.., ..x.., ..y.., ..ymin.., ..ymax..

ggplot() + stat_function(aes(x = -3:3), n = 99, fun = dnorm, args = list(sd=0.5)) **x** | ..x.., ..y..

e + stat_identity(na.rm = TRUE)

ggplot() + stat_qq(aes(sample=1:100), dist = qt, dparam=list(df=5)) **sample, x, y** | ..sample.., ..theoretical..

e + stat_sum() x, y, size | ..n.., ..prop..

e + stat_summary(fun.data = "mean_cl_boot")

h + stat_summary_bin(fun.y = "mean", geom = "bar")

e + stat_unique()

标尺

将映射数据缩放到较为美观的比例。添加新的标尺来改变映射。

(n <- d + geom_bar(aes(fill = fl)))

scale_开头 | 调整的参数 | 预设参数 | 标尺特定属性的参数

n + scale_fill_manual(
values = c("skyblue", "royalblue", "blue", "navy"),
limits = c("d", "e", "p", "r"), breaks =c("d", "e", "p", "r"),
name = "fuel", labels = c("D", "E", "P", "R"))

映射中包含值的范围 | 图例/轴中使用的标题 | 图例/轴中使用的标签 | 图例/轴的分隔

标尺的一般用法

scale_*_continuous() - 将数据的连续取值映射为图形属性的取值

scale_*_discrete() - 将数据的离散取值映射为图形属性的取值

scale_*_identity() - 使用数据的值作为图形属性的取值

scale_*_manual(values = c()) -将数据的离散取值作为手工指定的图形属性的取值

scale_*_date(date_labels = "%m/%d"), date_breaks = "2 weeks") - 将数据值视为日期

scale_*_datetime() - 将数据x视为时间

参数和scale_x_date()一样。有关标签格式请参阅striptime

调整X和Y的比例

调整x和y的标尺(使用x为例)
scale_x_log10() – 以log10比例绘制x
scale_x_reverse() -反转x轴方向
scale_x_sqrt() –以平方根绘制x

颜色和填充比例（离散）

n <- d + geom_bar(aes(fill = fl))
n + scale_fill_brewer(palette = "Blues")
选择调色板：: RColorBrewer::display.brewer.all()
n + scale_fill_grey(start = 0.2, end = 0.8, na.value = "red")

颜色和填充比例（连续）

o <- c + geom_dotplot(aes(fill = ..x..))
o + scale_fill_distiller(palette = "Blues")

o + scale_fill_gradient(low="red", high="yellow")

o + scale_fill_gradient2(low="red", high="blue", mid = "white", midpoint = 25)

o + scale_fill_gradientn(colours=topo.colors(6))
Also: rainbow(), heat.colors(), terrain.colors(), cm.colors(), RColorBrewer::brewer.pal()

形状和尺寸比例

p <- e + geom_point(aes(shape = fl, size = cyl))
p + scale_shape() + scale_size()
p + scale_shape_manual(values = c(3:7))

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

p + scale_radius(range = c(1,6))
p + scale_size_area(max_size = 6)

坐标系统

r <- d + geom_bar()

r + coord_cartesian(xlim = c(0, 5))
xlim, ylim
默认笛卡尔坐标系

r + coord_fixed(ratio = 1/2)
ratio, xlim, ylim
x和y单位之间固定长宽比的笛卡尔坐标

r + coord_flip()
xlim, ylim
翻转的笛卡尔坐标

r + coord_polar(theta = "x", direction=1)
theta, start, direction
极坐标

r + coord_trans(ytrans = "sqrt")
xtrans, ytrans, limx, limy
转换后的笛卡尔坐标。 将xtrans和ytrans设置为窗口函数的名称。

π + coord_quickmap()

π + coord_map(projection = "ortho", orientation=c(41, -74, 0))projection, orienztation, xlim, ylim
从mapproj包中映射投影(mercator (default), azequalarea, lagrange, etc.)

位置调整

位置调整决定了如何安排原本会占据相同空间的图例。

s <- ggplot(mpg, aes(fl, fill = drv))

s + geom_bar(position = "dodge")
并排排列元素

s + geom_bar(position = "fill")
堆叠元素并标准化高度

e + geom_point(position = "jitter")
将随机抖动添加到每个元素的x和Y位置以避免重叠

e + geom_label(position = "nudge")
标签稍远离数据点

s + geom_bar(position = "stack")
堆叠元素

每个位置调整都可以重新编写为具有手动宽度和高度参数的函数
s + geom_bar(position = position_dodge(width = 1))

主题

r + theme_bw()
网格白色背景

r + theme_gray()
灰色背景（默认主题）

r + theme_dark()
黑色背景

r + theme_classic()
r + theme_light()
r + theme_linedraw()

r + theme_minimal()
简单主题

r + theme_void()
清空主题

分面

根据一个或多个离散变量划分子图。

t <- ggplot(mpg, aes(cty, hwy)) + geom_point()

t + facet_grid(. ~ fl)
基于fl的列分面

t + facet_grid(year ~ .)
基于year的行分面

t + facet_grid(year ~ fl)
列和行的分面图

t + facet_wrap(~ fl)
包裹成矩形布局的分面图
设置scales限制分面坐标轴

t + facet_grid(drv ~ fl, scales = "free")
x和y轴适应各自的分面
"free_x" – 限制调整x轴
"free_y" - 限制调整y轴

设置labeller属性调整分面的标签

t + facet_grid(. ~ fl, labeller = label_both)

fl: c	fl: d	fl: e	fl: p	fl: r

t + facet_grid(fl ~ ., labeller = label_bquote(alpha ^ .(fl)))

α^{c} α^{d} α^{e} α^{p} α^{r}

t + facet_grid(. ~ fl, labeller = label_parsed)

c	d	e	p	r

标签

t + labs(**x** = "New x axis label", **y** = "New y axis label",
title ="Add a title above the plot",
subtitle = "Add a subtitle below title",
caption = "Add a caption below plot",
<AES> = "New <AES> legend title")
使用缩放功能更新图例标签
t + annotate(geom = "text", x = 8, y = 9, label = "A")

放置的位置 | 标签的数据

图例

n + theme(legend.position = "bottom")
放置图例： "bottom", "top", "left", or "right"

n + guides(fill = "none")
设置图例类型: colorbar, legend, or none (no legend)

n + scale_fill_discrete(name = "Title",
labels = c("A", "B", "C", "D", "E"))
使用scale函数设置图例标签

缩放

没有裁剪（推荐）
t + coord_cartesian(
xlim = c(0, 100), ylim = c(10, 20))

裁剪（删除看不见的数据点）
t + xlim(0, 100) **+ ylim**(10, 20)

t + scale_x_continuous(limits = c(0, 100)) **+ scale_y_continuous**(limits = c(0, 100))