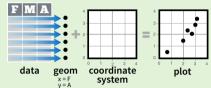
Визуализация данных при помощи ggplot2 Шпаргалка

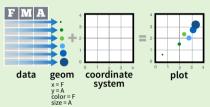


Основы

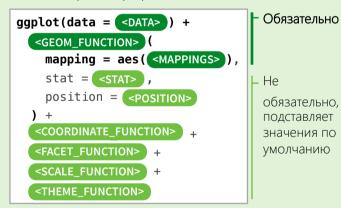
ggplot2 основан на "графической грамматике" идее, согласно которой любой график можно построить из одних и тех же составляющих: массива данных (data), системы координат и **геометрических объектов (geom)** — визуальных маркеров, отражающих частные значения наблюдений (точки, линии, столбцы и т.д.).



Значения наносятся на график при помощи визуальных свойств геометрического объекта (aesthetics), таких как размер, цвет, координаты х и



Чтобы построить график заполните этот шаблон:



ggplot(data = mpg, aes(x = cty, y = hwy))

Выводит пустой график, на который слой за слоем добавляются геометрические объекты.

qplot(x = cty, y = hwy, data = mpg, geom = "point")

Выводит полноценный график, с заданными вами данными, геометрией и разметкой. Множество полезных значений подставляются по умолчанию.

last_plot() Возвращает последний график.

ggsave("plot.png", width = 5, height = 5)

Сохраняет последний график в рабочий каталог как файл размером 5x5 дюймов с именем "plot.png" Сопоставляет формат файла с расширением.

Geoms - Используйте функцию geom чтобы нанести частные значения. Чтобы отобразить переменные используйте визуальные свойства.

Графические основы

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



(Полезно для расширения границ)



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



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



+ 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, xminymax, 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

Отрезки

Визуальные свойства: 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(vend=lat+1, xend=long+1))

b + geom_spoke(aes(angle = 1:1155, radius = 1)**)**

Одна переменная

Непрерывная

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



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



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



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



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



+ 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

Дискретная

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



geom bar()

x, alpha, color, fill, linetype, size, weight

$oldsymbol{\Delta}$ ве переменные

Непрерывная X, непрерываня Y e <- ggplot(mpg, aes(cty, hwy))



e + geom_label(aes(label = cty), nudge_x = 1, nudge y = 1, check overlap = TRUE) x. v. label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust



e + geom_point()

x, y, alpha, color, fill, shape, size, stroke

x, v, alpha, color, fill, shape, size



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

Дискретная Х, непрерываня Ү f <- ggplot(mpg, aes(class, hwy))



+ geom_col()

x, y, alpha, color, fill, group, linetype, size



geom_boxplot()





geom_dotplot(binaxis = "y", stackdir = "center")



x, y, alpha, color, fill, group + geom_violin(scale = "area")



x, y, alpha, color, fill, group, linetype, size,

Дискретная Х, дискретаня Ү

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



g + geom_count()

x, y, alpha, color, fill, shape, size, stroke

Двумерное нормальное распределение 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

Непрерывное отображение

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



i + geom_area()

x, y, alpha, color, fill, linetype, size



x, y, alpha, color, group, linetype, size

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

Визуализация ошибок

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))</pre>



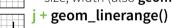
+ geom_crossbar(fatten = 2)

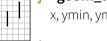
x, y, ymax, ymin, alpha, color, fill, group, linetype, size



+ geom errorbar()

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





x, ymin, ymax, alpha, color, group, linetype, size



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

data <- data.frame(murder = USArrests\$Murder, state = tolower(rownames(USArrests))) map <- map_data("state")</pre> 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

Три переменные

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



+ geom_contour(aes(z = z))

x, y, z, alpha, colour, group, linetype, size,



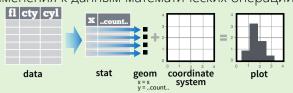
geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE) x, y, alpha, fill



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

Stats - Статистическое преобразование

Наносит на график новую переменную — результат применения к данным математических операций.



Отображение статистически преобразованной переменной: измените значение по умолчанию в геометрии: geom_bar(stat="count"); либо используя соответствующую статистику: stat count(geom="bar"). Чтобы разметить преобразованную переменную на графике используйте синтаксис типа ..name..



геометрия статистика визуальная схема

+ stat_density2d(aes(fill = ..level..),

geom = "polygon") новая переменная

- 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.. 1D распределения
- e + stat_bin_2d(bins = 30, drop = T) ^{2D} распределения 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.. 3 переменные

f + stat boxplot(coef = 1.5)

Сравнения

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..

Функции

 $e + stat_quantile(quantiles = c(0.1, 0.9),$

formula = $y \sim log(x)$, method = "rq") x, y | ..quantile..

 $e + stat_smooth(method = "lm", formula = y \sim x,$ 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()

Многоцелевые

Scales - Шкалы

Шкалы сопоставляют значения данных и их визуально воспринимаемое отображение (mapping).



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"))

Диапазон значений Подпись

Интервалы

Шкалы общего назначения

scale_*_continuous() - отобразить непрерывные величины

scale * discrete() - отобразить дискретн, величины scale * identity() - отобр. визуальные св-ва данных **scale_*_manual(**values = c()) - пользовательское

отображение дискретных величин scale *_date(date labels = "%m/%d"),

date_breaks = "2 weeks") - отображение дат scale_*_datetime() - отображение даты и времени (класс POSIXct). Аргументы, как для scale_x_date(). Смотри ?strptime для точной настройки меток.

Шкалы позиции X и Y (На примере x)

 $scale_x_log10()$ - Разметить x на логарифмической шкале с основанием 10 scale_x_reverse() - Обратить направление шкалы х scale_x_sqrt() - Шкала квадратных корней

Цвет и заливка (Дискретные) n <- d + geom bar(aes(fill = fl))



n + scale_fill_brewer(palette = "Blues") Для выбора палитры: RColorBrewer::display.brewer.all()

n + scale_fill_grey(start = 0.2, end = 0.8, na.value = "red")

Цвет и заливка (Непрерывные) 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()

Форма и размер

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 $\Box \circ \triangle + \times \Diamond \nabla \boxtimes \# \oplus \triangle \boxtimes \boxtimes \boxtimes \blacksquare \bullet \blacktriangle \bullet \bullet \circ \Box \Diamond \triangle \nabla$



 $p + scale_radius(range = c(1,6))$ p + scale_size_area(max size = 6)

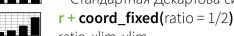
Системы координат

r <- d + geom bar()



r + coord cartesian(xlim = c(0, 5))xlim. vlim

Стандартная Декартова система



ratio, xlim, ylim

Декартова система с фиксированными пропорциями между величинами х и у



xlim, ylim

Перевёрнутая Декартова система



r + coord polar(theta = "x", direction=1) theta, start, direction

Полярная система координат



r + coord_trans(ytrans = "sqrt") xtrans, ytrans, limx, limy Изменённая Декартова система. Задайте xtrans и ytrans для преобразования значений.

 π + coord quickmap()

 π + coord map(projection = "ortho", orientation=c(41, -74, 0))

projection, orientation, xlim, ylim

Картографические проекции из пакета тарргоі (mercator (умолчание), azequalarea, lagrange, и т.д.)

Facets - Панели

Панели делят график на подграфики на основе значений одной или нескольких дискретных переменных.

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



разбить на строки на основе vear t + facet grid(year ~ fl)

разбить и на строки и на столбцы

t + facet wrap(~ fl)

прямоугольная схема разбиения

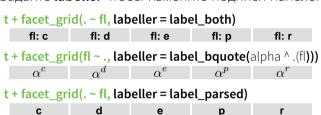
Чтобы установить индивидуальные диапазоны осей задайте аргумент scales

t + facet grid(dry ~ fl, scales = "free")

Лимиты осей х и у регулируются индивидуально

- "free_x" регулировка границ оси x
- "free_y" регулировка границ оси у

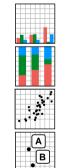
Задайте **labeller** чтобы изменить подписи панелей



Корректировка позиции

Позволяет перераспределить перекрывающие друг друга геометрические объекты.

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



s + geom_bar(position = "dodge") Распределяет элементы бок о бок

s + geom_bar(position = "fill") Укладывает элементы поверх друг

друга, нормализует высоту столбцов e + geom_point(position = "jitter") Добавляет случайный шум к позициям Х

и Y для каждокго элемента e + geom_label(position = "nudge")

Отдаляет метки подальше от точек s + geom_bar(position = "stack")

Укладывает элементы поверх друг друга Любая регулировка позиции может быть

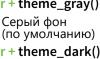
переквалифицирована как функция с ручной регулировкой ширины и высоты.

s + geom_bar(position = position_dodge(width = 1))

Темы



Чёрно-белыая ема



- r + theme_light() r + theme linedraw() r + theme_minimal() Минималистичные

r + theme_classic()

r + theme_void() Пустая тема

Метки

t + labs(x = "Метка оси x", y = "Метка оси y",title = "Добавляет заголовок над графиком",

subtitle = "Добавляет подзаголовок",

caption = "Подпись под графиком", <AES> = "Новый заголовок метки <AES> ")

t + annotate(geom = "text", x = 8, y = 9, label = "A")

геометрия

вручную заданные значения для визуальной схемы

Легенда

n + theme(legend.position = "bottom") Регулировка позиции легенды

n + guides(fill = "none")

Устанавливает разный тип легенды для каждой визуальной схемы: colorbar, legend, или none

n + scale fill discrete(name = "Title", labels = c("A", "B", "C", "D", "E")**)**

Заголовок легенды и метки задать через ф-ю scale.

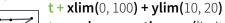
Масштабирование



Без обрезки (предпочтительно) t + coord cartesian(

 $x \lim = c(0, 100), y \lim = c(10, 20)$

С обрезкой (обрезает знаения вне поля зрения)





t + scale x continuous(limits = c(0, 100)) +scale_y_continuous(limits = c(0, 100))

· + theme_gray()

theme_dark() Тёмная тема