

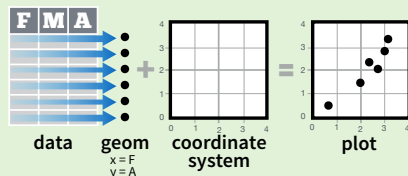
Data Visualization with ggplot2

Cheat Sheet

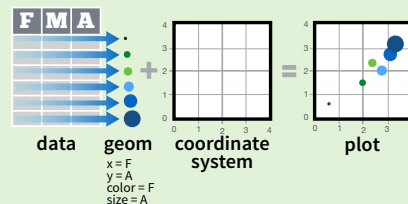


Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.



Complete the template below to build a graph.

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

Required

Not required, sensible defaults supplied

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

Begins a plot that you finish by adding layers to. Add one geom function per layer.

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

Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last_plot()

Returns the last plot

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

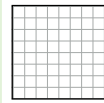
Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Geoms - Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

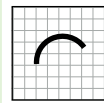
Graphical Primitives

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

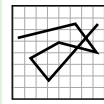
b <- ggplot(seals, aes(x = long, y = lat))



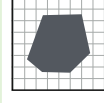
a + geom_blank()
(Useful for expanding limits)



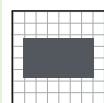
b + geom_curve(aes(yend = lat + 1, xend = long + 1, curvature = z)) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size



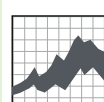
a + geom_path(lineend = "butt", linejoin = "round", linemitre = 1) - x, y, alpha, color, group, linetype, size



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



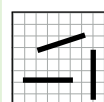
b + geom_rect(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size



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

Line Segments

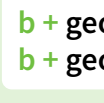
common aesthetics: x, y, alpha, color, linetype, size



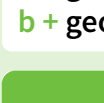
b + geom_abline(aes(intercept = 0, slope = 1))

b + geom_hline(aes(yintercept = lat))

b + geom_vline(aes(xintercept = long))



b + geom_segment(aes(yend = lat + 1, xend = long + 1))

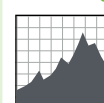


b + geom_spoke(aes(angle = 1:1155, radius = 1))

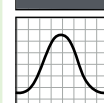
One Variable

Continuous

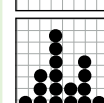
c <- ggplot(mpg, aes(hwy)); **c2** <- ggplot(mpg)



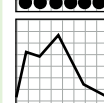
c + geom_area(stat = "bin") - x, y, alpha, color, fill, linetype, size



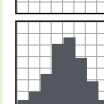
c + geom_density(kernel = "gaussian") - x, y, alpha, color, fill, group, linetype, size, weight



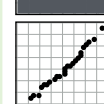
c + geom_dotplot() - x, y, alpha, color, fill



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



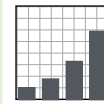
c + geom_histogram(binwidth = 5) - x, y, alpha, color, fill, linetype, size, weight



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

Discrete

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

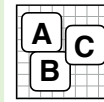


d + geom_bar() - x, alpha, color, fill, linetype, size, weight

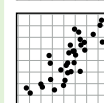
Two Variables

Continuous X, Continuous Y

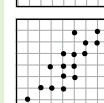
e <- ggplot(mpg, aes(cty, hwy))



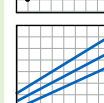
e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust



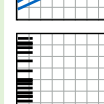
e + geom_jitter(height = 2, width = 2) - x, y, alpha, color, fill, shape, size



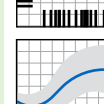
e + geom_point() - x, y, alpha, color, fill, shape, size, stroke



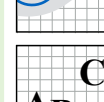
e + geom_quantile() - x, y, alpha, color, group, linetype, size, weight



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



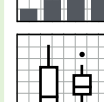
e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

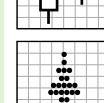
f <- ggplot(mpg, aes(class, hwy))



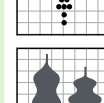
f + geom_col() - x, y, alpha, color, fill, group, linetype, size



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



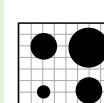
f + geom_dotplot(binaxis = "y", stackdir = "center") - x, y, alpha, color, fill, group



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

Discrete X, Discrete Y

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



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

Three Variables

seals\$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))

l <- ggplot(seals, aes(long, lat))



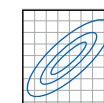
l + geom_contour(aes(z = z)) - x, y, z, alpha, colour, group, linetype, size, weight

Continuous Bivariate Distribution

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



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



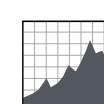
h + geom_density2d() - x, y, alpha, colour, group, linetype, size



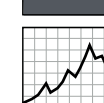
h + geom_hex() - x, y, alpha, colour, fill, size

Continuous Function

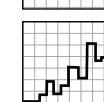
i <- ggplot(economics, aes(date, unemploy))



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



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

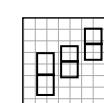


i + geom_step(direction = "hv") - x, y, alpha, color, group, linetype, size

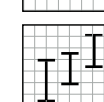
Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)

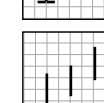
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))



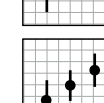
j + geom_crossbar(fatten = 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size



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



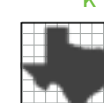
j + geom_pointrange() - x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

Maps

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

