Leiden University
Medical Center

Basics of data manipulation and visualization of biological data in R

Dr. Quinten Ducarmon, Dr. Morgan Essex & Prof. Georg Zeller

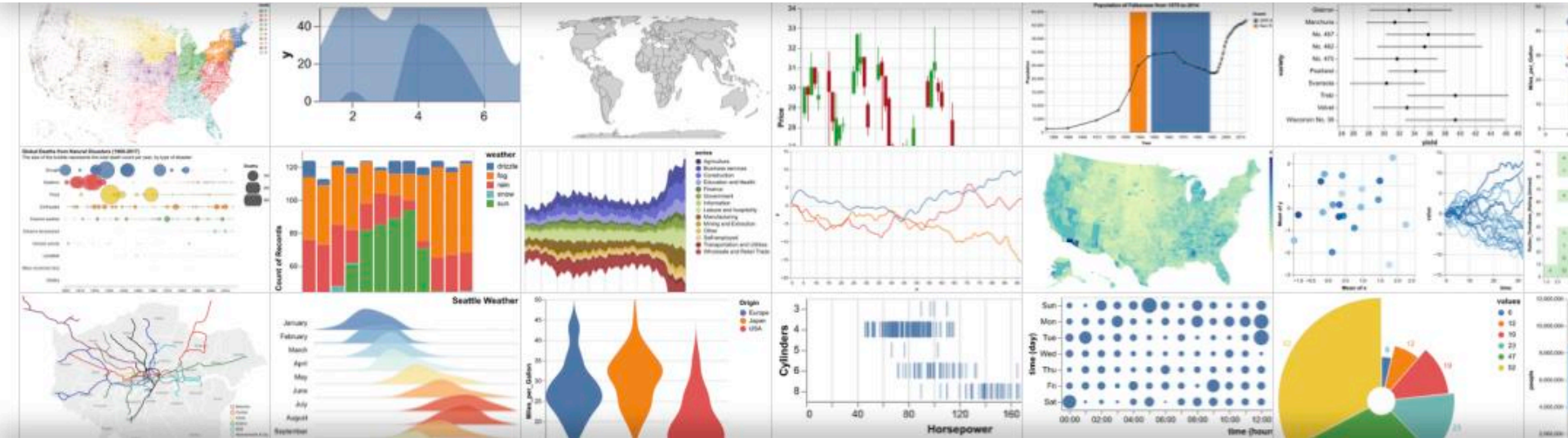
Quinten Ducarmon

IAI Course

LEIDEN, OCTOBER 24TH 2025

Learning objectives of today's lecture

- Refresh summary statistics and learn to calculate and visualize them in R
- Understand a case-control study design
- Understand fundamental visualizations and their pros/cons
- Gain simple building blocks that can still get you very far



Refresher from Day 1

```
1  # variables and function syntax
2  z <- c(1, 2, 5, 8)
3  z_mean <- mean(z)

4  # data frames, pipes, simple plots
5  df <- mtcars |>
6    rownames_to_column(var = "model") |>
7    as_tibble()

8  df |>
9    ggplot(aes(x = mpg)) +
10   geom_histogram(bins = 20) +
11   labs(title = "Distribution of MPG")
```

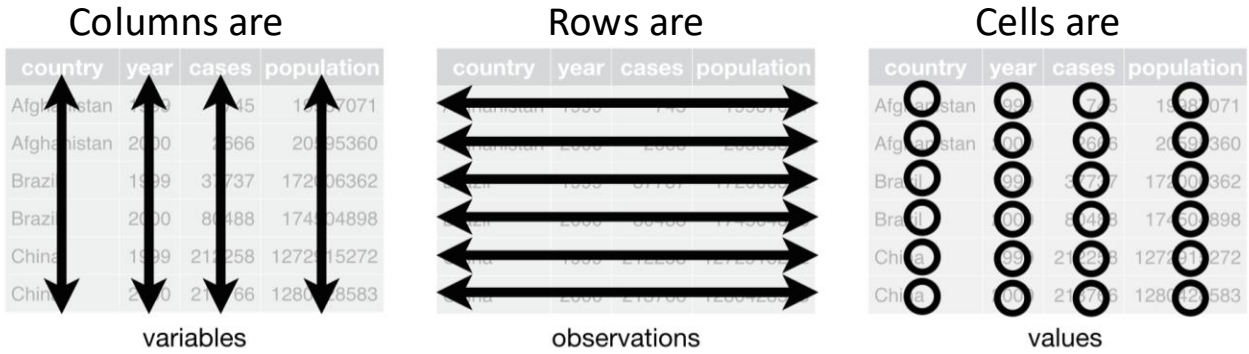
Human clinical and health data

Data: from the lab / clinic to your computer

- Primary biomedical (e.g. count) data often comes in a **predictable format**
- However, patient and study metadata are often **heterogeneous and messy**
- **Cleaning data** is a prerequisite for statistical analysis

- "**Tidy**" refers a specific set of principles for cleaning and transforming **tabular data**
- Becomes essential for predictive nature of data science (vs. analysis)

Data is tidy when…



Example tidy and untidy data

Which is which?

```
> gapminder.long
# A tibble: 1,704 x 3
   country     year  gdpPercap
   <fct>       <chr>     <dbl>
 1 Afghanistan 1952       779.
 2 Afghanistan 1957       821.
 3 Afghanistan 1962       853.
 4 Afghanistan 1967       836.
 5 Afghanistan 1972       740.
 6 Afghanistan 1977       786.
 7 Afghanistan 1982       978.
 8 Afghanistan 1987       852.
 9 Afghanistan 1992       649.
10 Afghanistan 1997       635.
# … with 1,694 more rows
```

```
> gapminder.wide
# A tibble: 142 x 13
   country     `1952` `1957` `1962` `1967` `1972` `1977` `1982` `1987` `1992` `1997` `2002` `2007`
   <fct>        <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
 1 Afghanistan   779.   821.   853.   836.   740.   786.   978.   852.   649.   635.   727.   975.
 2 Albania      1601.  1942.  2313.  2760.  3313.  3533.  3631.  3739.  2497.  3193.  4604.  5937.
 3 Algeria      2449.  3014.  2551.  3247.  4183.  4910.  5745.  5681.  5023.  4797.  5288.  6223.
 4 Angola       3521.  3828.  4269.  5523.  5473.  3009.  2757.  2430.  2628.  2277.  2773.  4797.
 5 Argentina    5911.  6857.  7133.  8053.  9443. 10079.  8998.  9140.  9308. 10967.  8798. 12779.
 6 Australia   10040. 10950. 12217. 14526. 16789. 18334. 19477. 21889. 23425. 26998. 30688. 34435.
 7 Austria      6137.  8843. 10751. 12835. 16662. 19749. 21597. 23688. 27042. 29096. 32418. 36126.
 8 Bahrain      9867. 11636. 12753. 14805. 18269. 19340. 19211. 18524. 19036. 20292. 23404. 29796.
 9 Bangladesh    684.   662.   686.   721.   630.   660.   677.   752.   838.   973.  1136.  1391.
10 Belgium      8343.  9715. 10991. 13149. 16672. 19118. 20980. 22526. 25576. 27561. 30486. 33693.
# … with 132 more rows
```

Is the microbiome count table tidy?

	Species 1	Species 2	Species 3	Species 4
Sample 1	38	60	77	205
Sample 2	25	55	86	82
Sample 3	18	19	22	65

In the practical part, we will explore different representations, and in which context one or the other is more useful…

Basic data structures in R

Vectors
Any number of elements
(of same type)

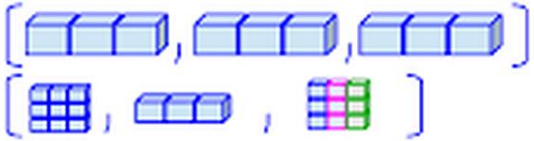
1 data type

Matrices
Homogeneous (usually numeric)
data organized in rows and columns

"1D"

"2D"

Lists
Any number of (heterogeneous) elements
Recursive (list of lists)

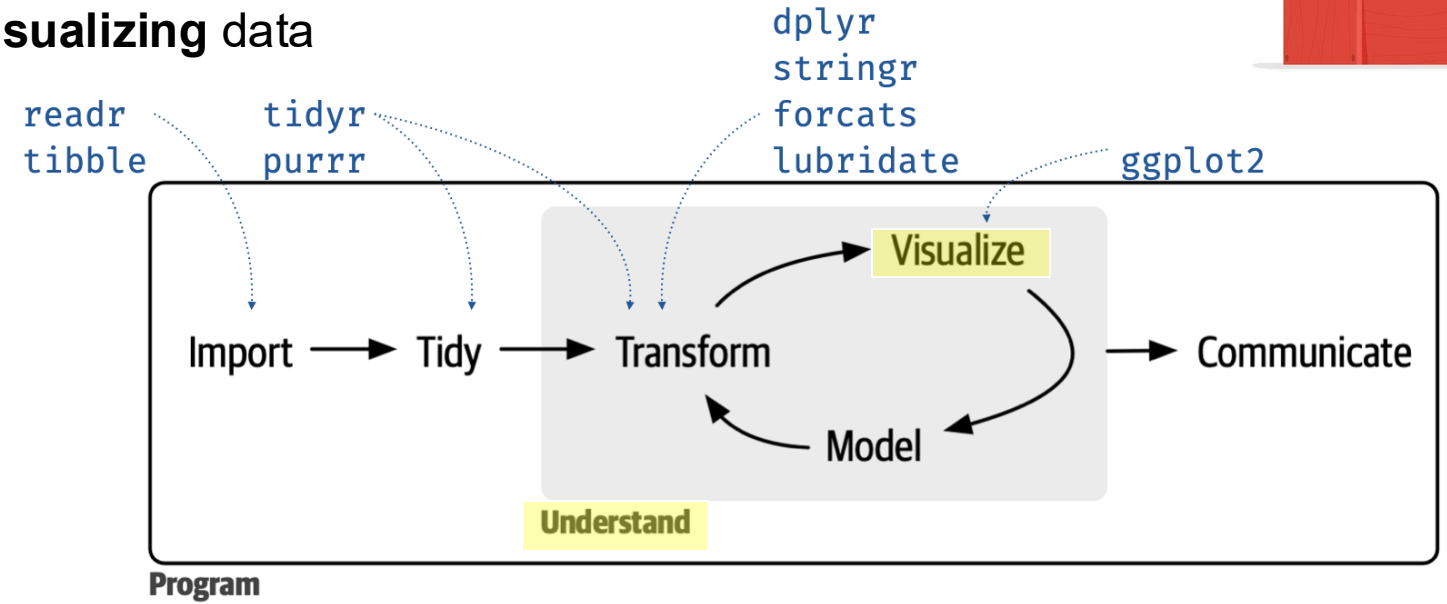


Data Frames
List of same-length vectors

1+ data types

library(tidyverse) as a complete data science toolbox

- A coherent set of "**packages**" (software libraries) for working with data in R
- This course: **understanding** and especially **visualizing** data



Counts and frequencies for categorical data

- Purpose

	name	score	subject	grade
1	Krunal	85	Math	10th
2	Ankit	90	Math	11th
3	Rushabh	78	History	11th
4	Niva	95	History	10th
5	Hemang	80	Math	11th
6	Vidisha	92	History	10th

r-lang.com

```
df_students_grade %>% count(subject)
```

	subject	n
1	History	3
2	Math	3

3 rows with History subject and 3 rows with Math subject

Summary statistics for metric data

- Purpose

df_students_grade

	name	score	subject	grade
1	Krunal	85	Math	10th
2	Ankit	90	Math	11th
3	Rushabh	78	History	11th
4	Niva	95	History	10th
5	Hemang	80	Math	11th
6	Vidisha	92	History	10th

r-lang.com

```
df_students_grade %>% dplyr::count(subject, wt = score, name = "total_score")
```

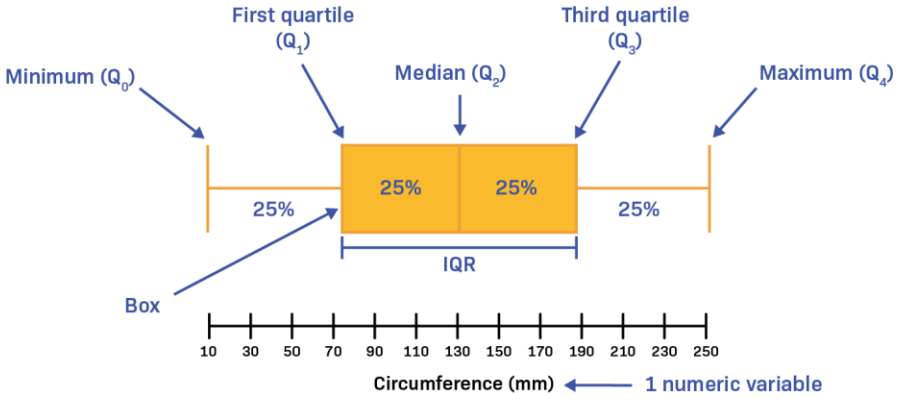
	subject	total_score	
1	History	265	78 + 95 + 92 = 265
2	Math	255	85 + 90 + 80 = 255

Summary statistics for metric data

- Purpose: simplify complex datasets into understandable numbers, identify patterns, facilitate comparison, aid decision-making

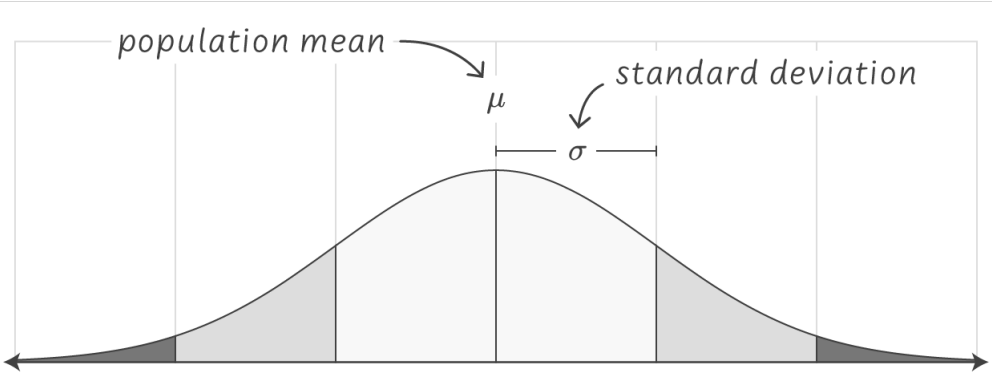
- Measures of central tendency
  - Mean, median, mode

- Measures of dispersion / spread
  - Range, standard deviation, quantiles / quartiles, interquartile range (IQR)



Summary statistics in R

- Default base R functions, e.g. `mean()` and `sd()` (standard deviation)
- The `summary()` function calculates several values at once
- Also works on an entire data frame --> “vectorized” command



$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

Anatomy of a ggplot

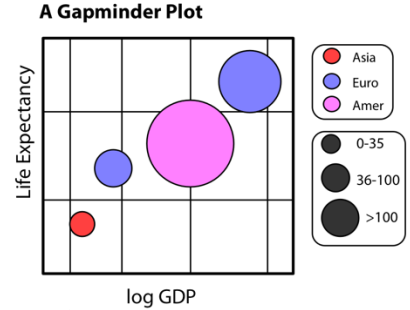
- Plots are built “bottom up”, *usually but not always starting with tidy data*


```
p <- ggplot(data = gapminder,
            mapping = aes(x = gdp,
                          y = lifexp,
                          color = continent)) +
  geom_point()
```

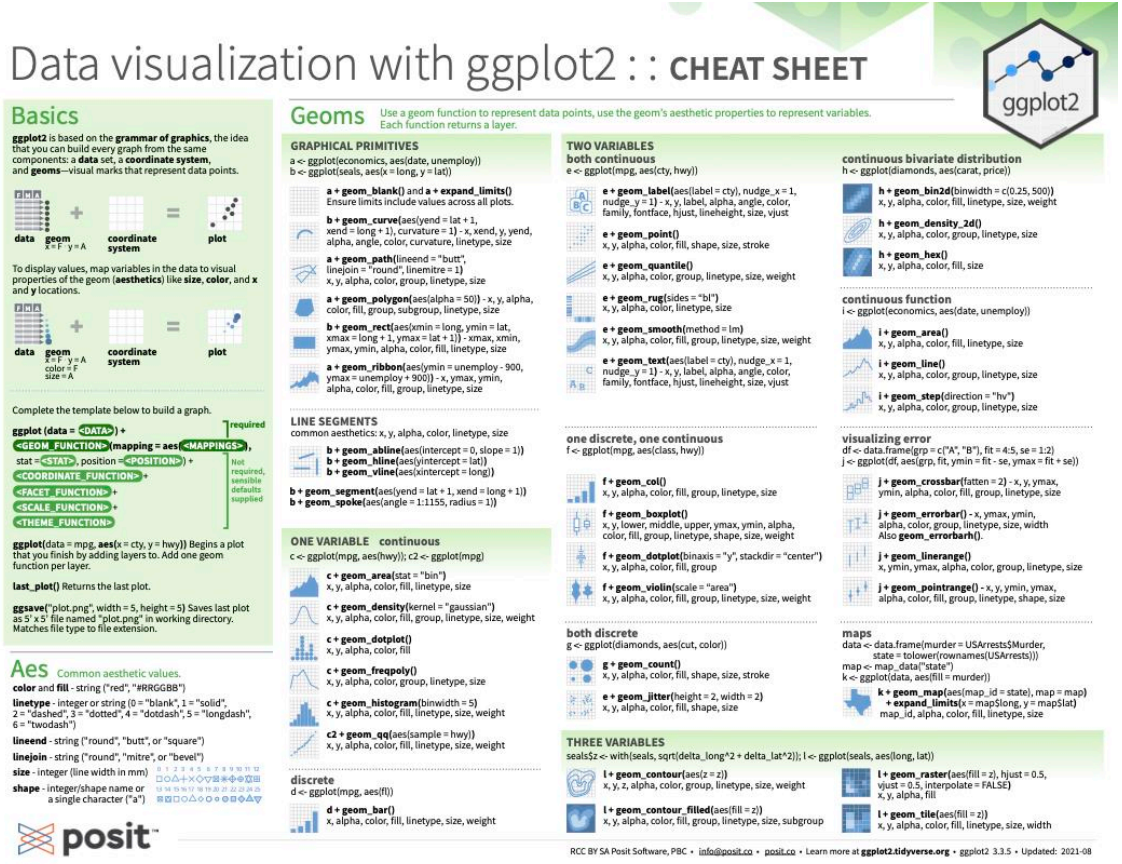
```
p ← ggplot(data = gapminder, ...
```

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

```
p + labs(x = “log GDP”, y = “Life
Expectancy”, title = “A Gapminder Plot”)
```



The tidyverse has free cheat sheets



Data visualization with ggplot2 : : CHEAT SHEET

ggplot2

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.

data + geom (x = F, y = A) = coordinate system + plot

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

data + geom (x = F, y = A, color = F, size = A) = coordinate system + plot

Complete the template below to build a graph.

required

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

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.

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.

Aes

Common aesthetic values.

color and **fill** - string ("red", "#RRGGBB")

linetype - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")

lineend - string ("round", "butt", or "square")

linejoin - string ("round", "mitre", or "bevel")

size - integer (line width in mm)

shape - integer/shape name or a single character ("a")

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() and **a + expand_limits()** Ensure limits include values across all plots.

b + geom_curve(aes(yend = lat + 1, xend = long + 1), curvature = 1) - 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(alpha = 50)) - x, y, alpha, color, fill, group, subgroup, 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

both continuous

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

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

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) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

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_density_2d() x, y, alpha, color, group, linetype, size

h + geom_hex() x, y, alpha, color, 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

one discrete, one continuous

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

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, width Also **geom_errorbarh()**.

j + geom_errorbar() - x, ymax, ymin, alpha, color, group, linetype, size, width

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

both discrete

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

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

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

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, color, group, linetype, size, weight

l + geom_contour_filled(aes(fill = z)) x, y, alpha, color, fill, group, linetype, size, subgroup

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

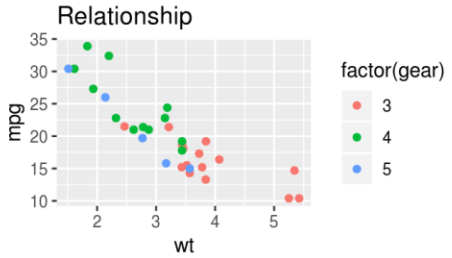
l + geom_tile(aes(fill = z)) x, y, alpha, color, fill, linetype, size, width

posit

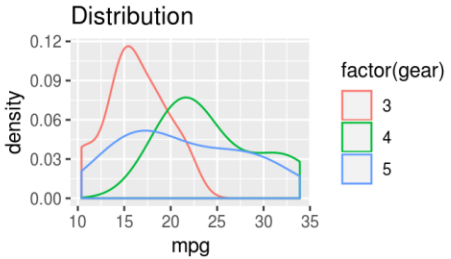
https://posit.co/wp-content/uploads/2022/10/data-visualization-1.pdf

4 essential functions of data visualization

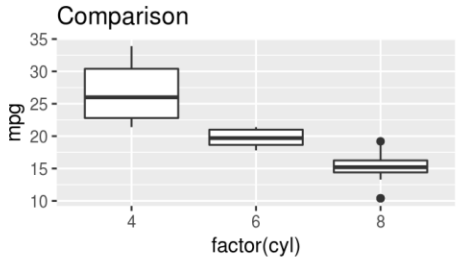
- Plot options differ depending on data type
- Multiple (≥1) options for each category



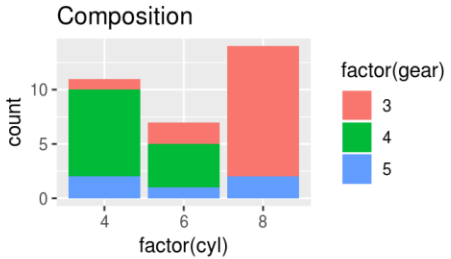
geom_point()



geom_density()



geom_boxplot()



geom_bar()

```
> data("mtcars")
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

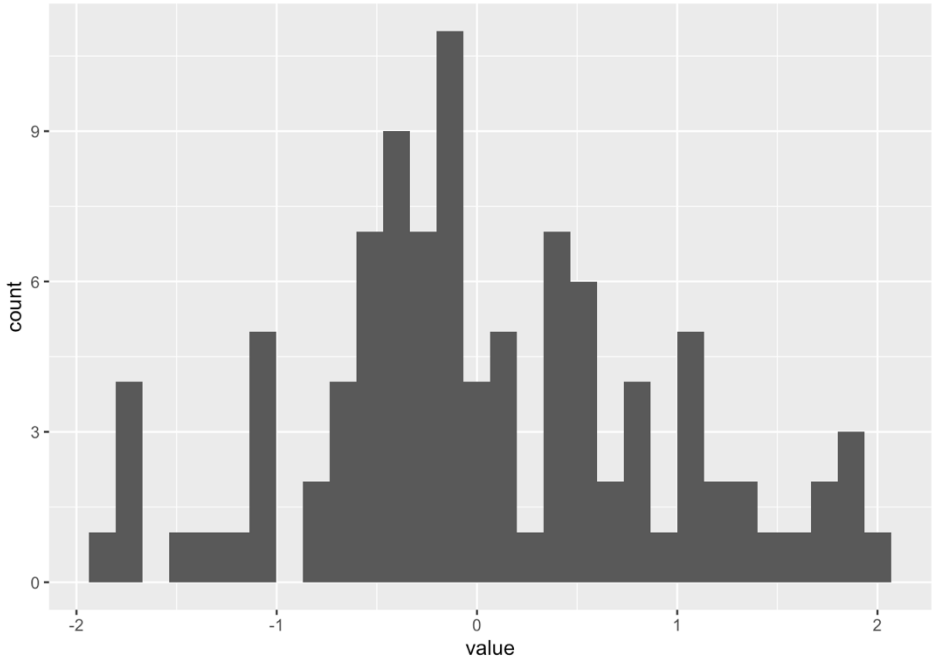
`mtcars` is a built-in dataset

`gapminder` is another

Basic visualization – histogram

- Visual representation of the distribution of quantitative data
- Requires only 1 numeric variable

```
# basic histogram
p <- ggplot(data, aes(x=value)) +
  geom_histogram()
```



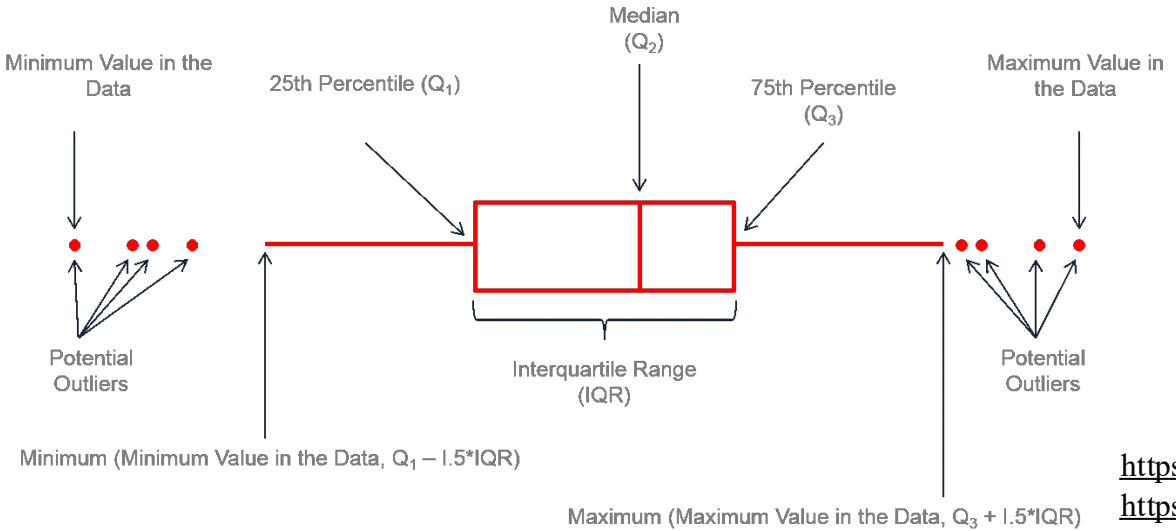
https://r-graph-gallery.com/

Basic visualization – boxplot

- Summarizes the distribution of quantitative data

```
# Basic boxplot
ggplot(mtcars, aes(x=as.factor(cyl), y=mpg)) +
    geom_boxplot()
```

- Quantitative variable for y-axis, qualitative variable for x-axis



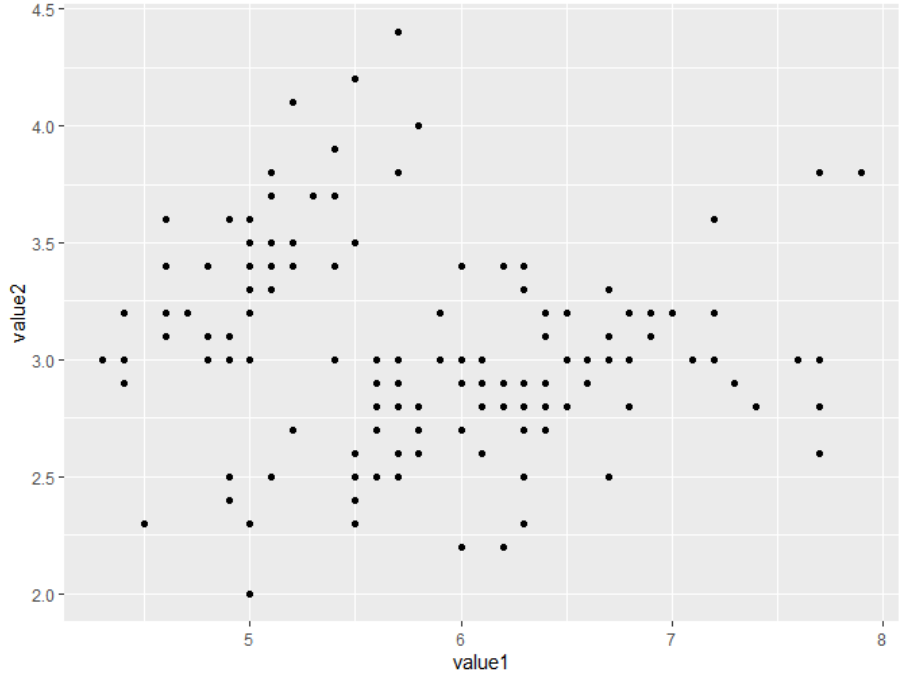
https://r-graph-gallery.com/
https://leansigmacorporation.com/box-plot-with-minitab/

Basic visualization – scatterplot

- Shows the relationship between two quantitative variables

```
# basic scatterplot

ggplot(iris, aes(x=value1, y=value2) +
    geom_point()
```

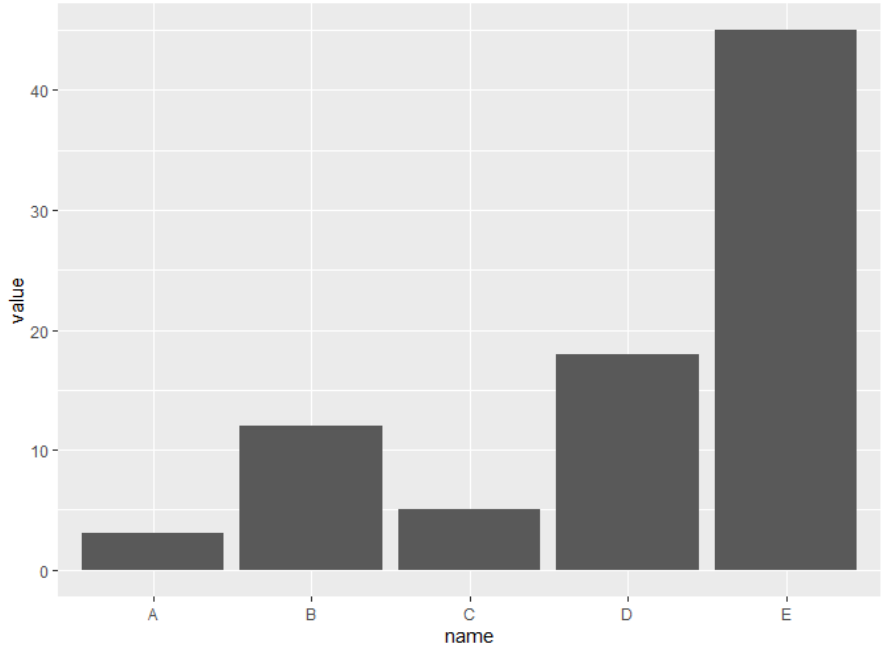


https://r-graph-gallery.com/

Basic visualization – barplot

- Shows the relationship between a quantitative and qualitative variable

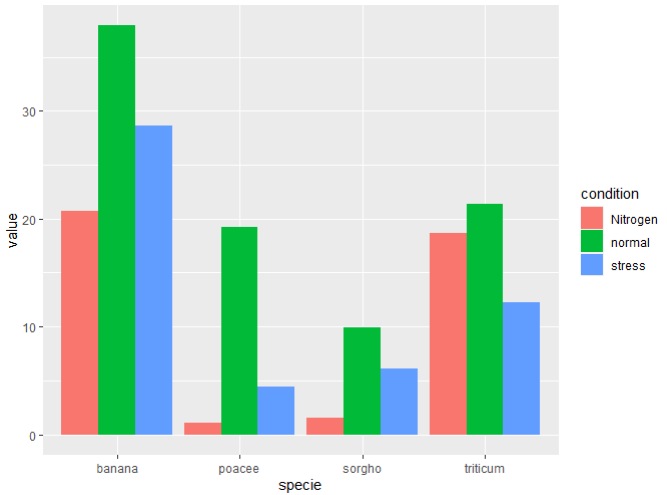
```
# Barplot
ggplot(data, aes(x=name, y=value)) +
  geom_bar(stat = "identity")
```

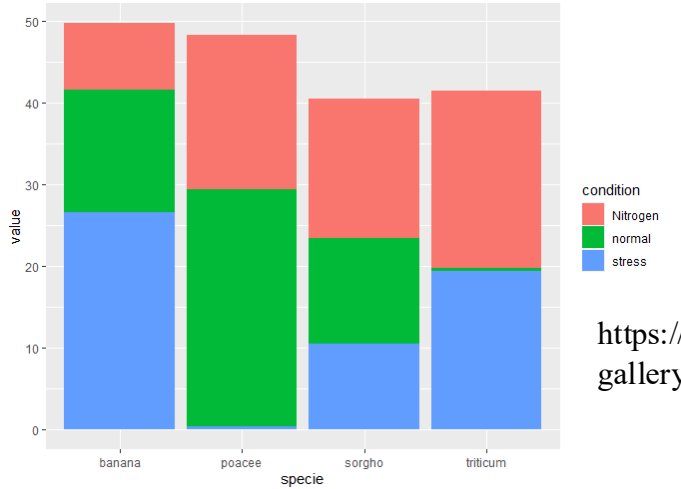


https://r-graph-gallery.com/

Basic visualization – Grouped & stacked bar plot

- Shows the contribution of different qualitative variables to a total value





https://r-graph-gallery.com/

```
# Grouped
ggplot(data, aes(fill=condition, y=value, x=specie)) +
    geom_bar(position="dodge", stat="identity")
```

```
# Stacked
ggplot(data, aes(fill=condition, y=value, x=specie)) +
    geom_bar(position="stack", stat="identity")
```