Effective Graphs with ggplot2

Beginner short course, SDSS 2023 (St. Louis)
May 23, 2023

Joyce Robbins
Dept. of Statistics, Columbia University
jtr13@columbia.edu

Schedule

- 8:30 - 9:30 grammar of graphics, histograms, density curves
- 9:30 - 10:00 lab
- 10:00 - 10:30 boxplots, scatterplots
- 10:30 - 10:45 lab
- 10:45 - 11:05 BREAK
- 11:05 - 11:15 lab
- 11:15 - 12:00 grammar of graphics, pivoting data frames
- 12:00 - 12:30 lab

Code and slides

www.github.com/jtr13/sdss2023

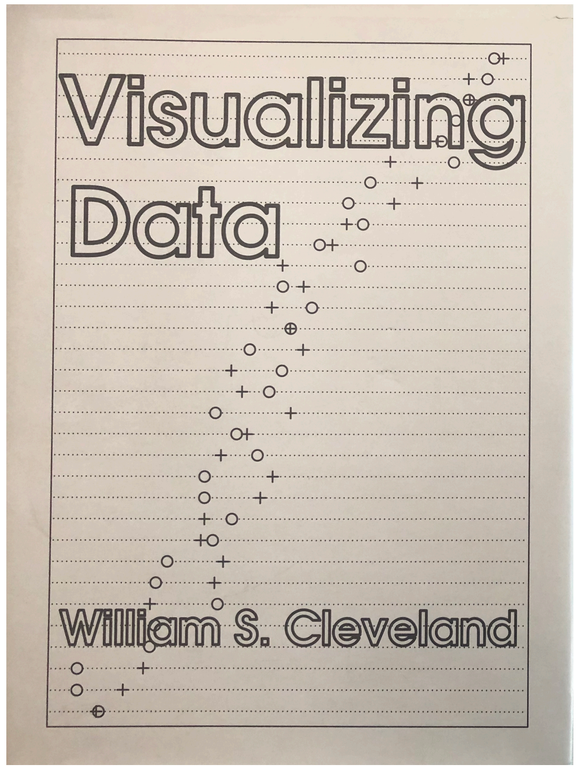
Why effective graphs?

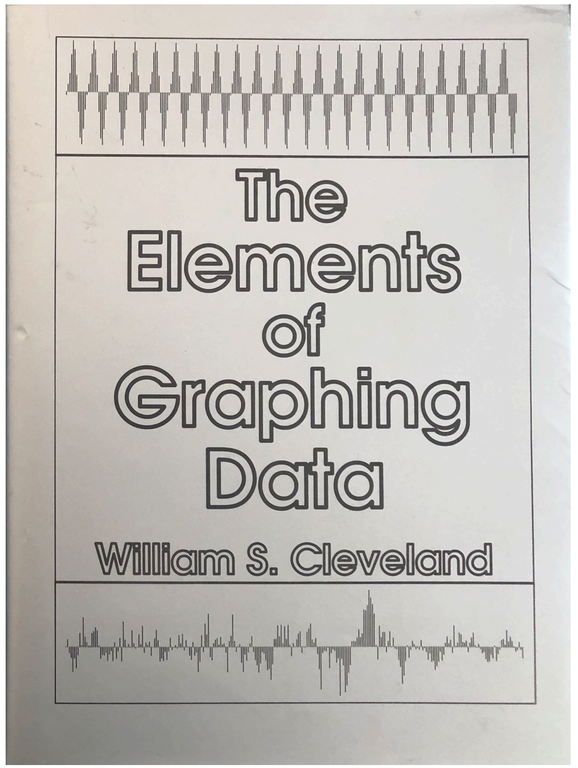
- Goal: discover and communicate trends in data
- "There is little evidence that the quality of the best graphics has improved over the last 100 years. I wonder if technology serves primarily as a **quantity**-multiplier, rather than a **quality**-multiplier." -Hadley Wickham
- Based on research on human perception
- Tight link between the development of S (precursor to R) and these studies

S, R, and graphics

- developed in the 1970s at Bell Labs as a system "for organizing, visualizing, and analyzing data"
- main goal: to create an interactive environment for statisticians using the most advanced analytical tools
- influenced by John Tukey's work on exploratory data analysis
- importance of statistical perspective / graphics research is still a defining feature of R today

William Cleveland





R help example

pie {graphics} R Documentation

Pie Charts

Description

Draw a pie chart.

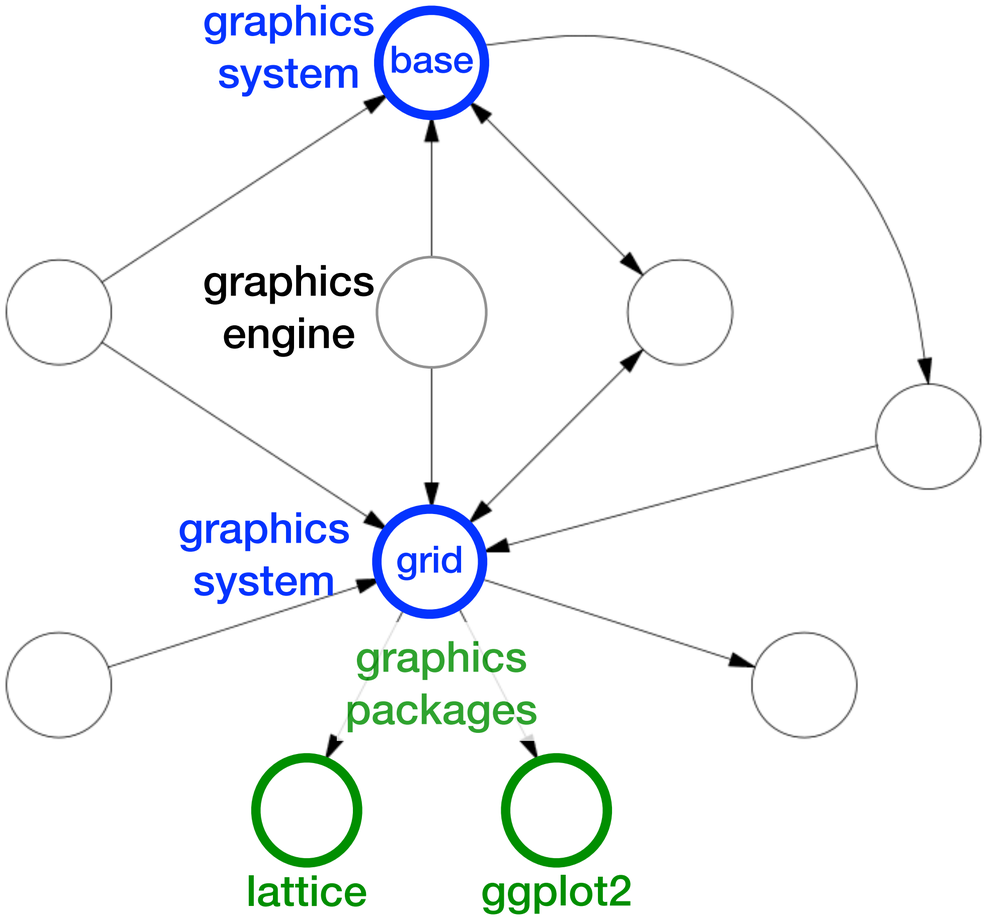
Usage

```
pie(x, labels = names(x), edges = 200, radius = 0.8,
    clockwise = FALSE, init.angle = if(clockwise) 90 else 0,
    density = NULL, angle = 45, col = NULL, border = NULL,
    lty = NULL, main = NULL, ...)
```

Arguments

`x`	a vector of non-negative numerical quantities. The values in `x` are displayed as the areas of pie slices.
`labels`	one or more expressions or character strings giving names for the slices. Other objects are coerced by `as.graphicsAnnot`. For empty or `NA` (after coercion to character) labels, no label nor pointing line is drawn.
`edges`	the circular outline of the pie is approximated by a polygon with this many edges.
`radius`	the pie is drawn centered in a square box whose sides range from *-1* to *1*. If the character strings labeling the slices are long it may be necessary to use a smaller radius.
`clockwise`	logical indicating if slices are drawn clockwise or counter clockwise (i.e., mathematically positive direction), the latter is default.
`init.angle`	number specifying the *starting angle* (in degrees) for the slices. Defaults to 0 (i.e., '3 o'clock') unless `clockwise` is true where `init.angle` defaults to 90 (degrees), (i.e., '12 o'clock').
`density`	the density of shading lines, in lines per inch. The default value of `NULL` means that no shading lines are drawn

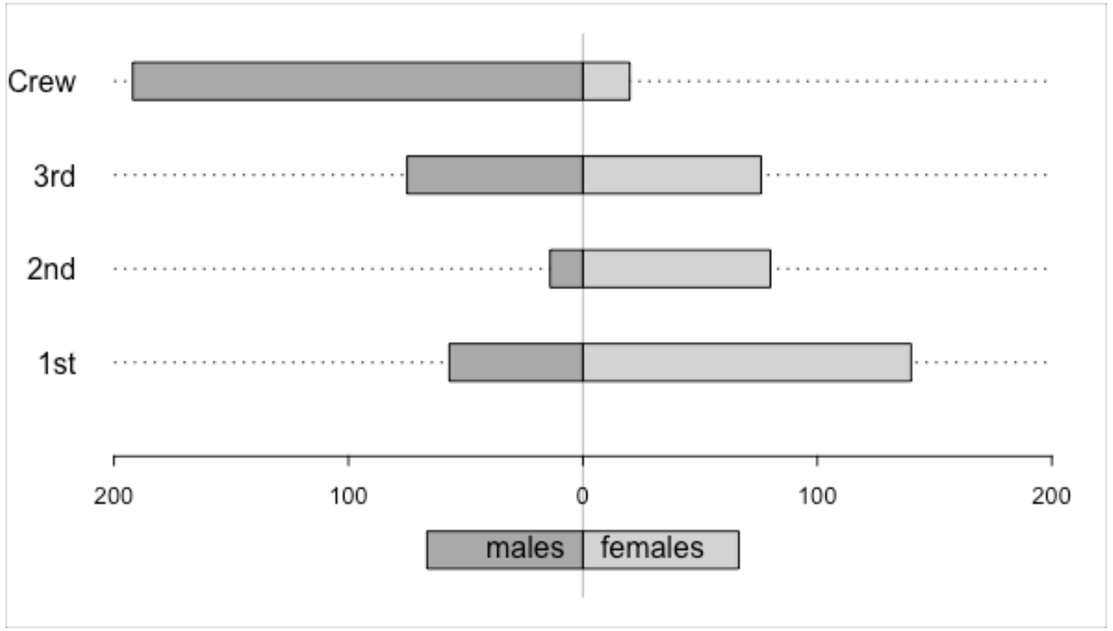
R graphics



Based on https://www.stat.auckland.ac.nz/~paul/RG3e/organisation-graphicslevels.png

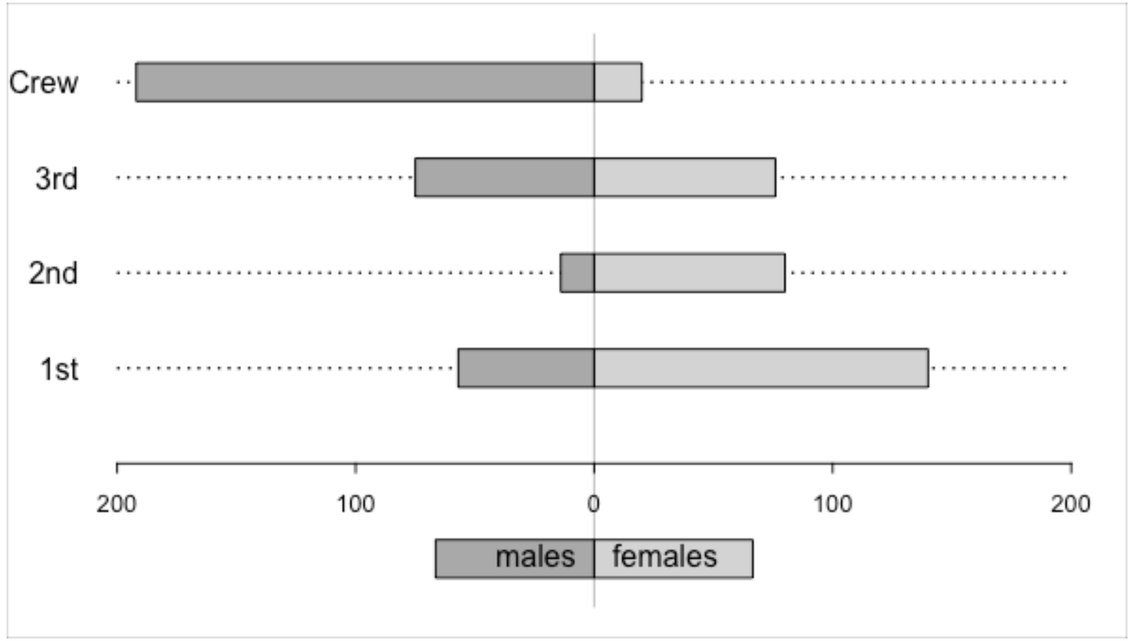
Base R graphics

```
groups <- dimnames(Titanic)[[1]]
males <- Titanic[, 1, 2, 2]
females <- Titanic[, 2, 2, 2]
par(mar=c(0.5, 4, 0.5, 1))
plot.new()
plot.window(xlim=c(-200, 200), ylim=c(-1.5, 4.5))
ticks <- seq(-200, 200, 100); y <- 1:4; h <- 0.2
lines(rep(0, 2), c(-1.5, 4.5), col="gray")
segments(-200, y, 200, y, lty="dotted")
rect(-males, y-h, 0, y+h, col="dark gray")
rect(0, y-h, females, y+h, col="light gray")
mtext(groups, at=y, adj=1, side=2, las=2)
par(cex.axis=0.8, mex=0.5)
axis(1, at=ticks, labels=abs(ticks), pos=0)
tw <- 1.5*strwidth("females")
rect(-tw, -1-h, 0, -1+h, col="dark gray")
rect(0, -1-h, tw, -1+h, col="light gray")
text(0, -1, "males", pos=2)
text(0, -1, "females", pos=4)
box("inner", col="gray")
```



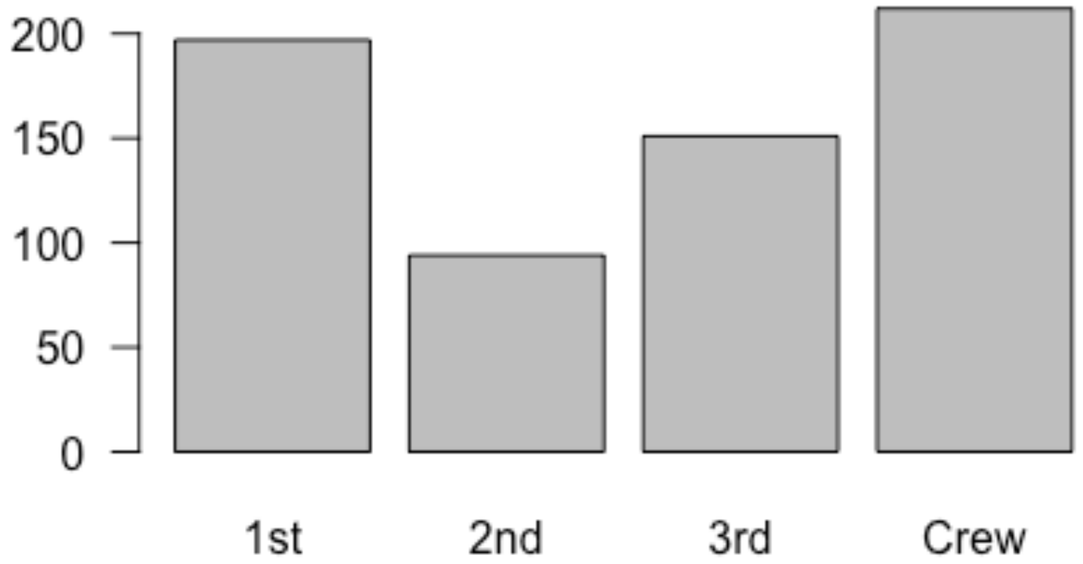
https://www.stat.auckland.ac.nz/~paul/RG3e/custombase-xmastree.R

Base R graphics



Higher level base R graphics functions

```
1 crew_counts <- rowSums(Titanic[,1:2,2,2])
2 barplot(crew_counts, las = 1)
```

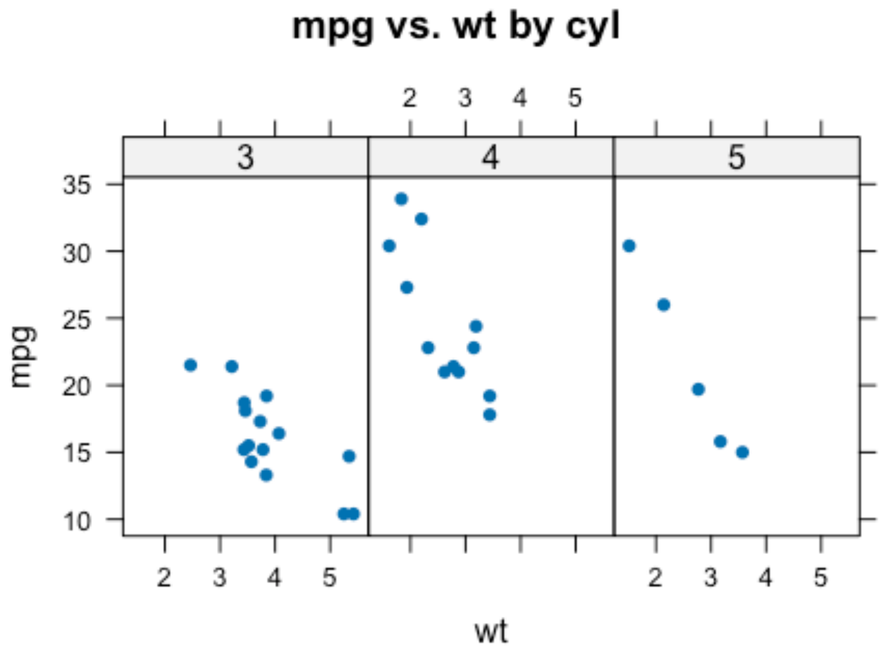


Higher level base R graphics functions

```
barplot()
boxplot()
cdplot()
contour()
coplot()
dotplot()
fourfoldplot()
hist()
matplot()
mosaicplot()
pairs()
pie()
plot()
smoothScatter()
spineplot()
stars()
stem()
stripchart()
sunflowerplot()
```

`lattice` package

```
1 library(lattice)
2 xyplot(mpg~wt|factor(cyl), data = mtcars,
3        main="mpg vs. wt by cyl", pch = 16)
```



`lattice` higher level plotting functions

```
xyplot()
splom()
cloud()
stripplot()
bwplot()
dotplot()
barchart()
histogram()
densityplot
qqmath()
qq()
contourplot()
levelplot()
parallel()
wireframe()
```

http://www.sthda.com/english/wiki/lattice-graphs

So... why ggplot2?

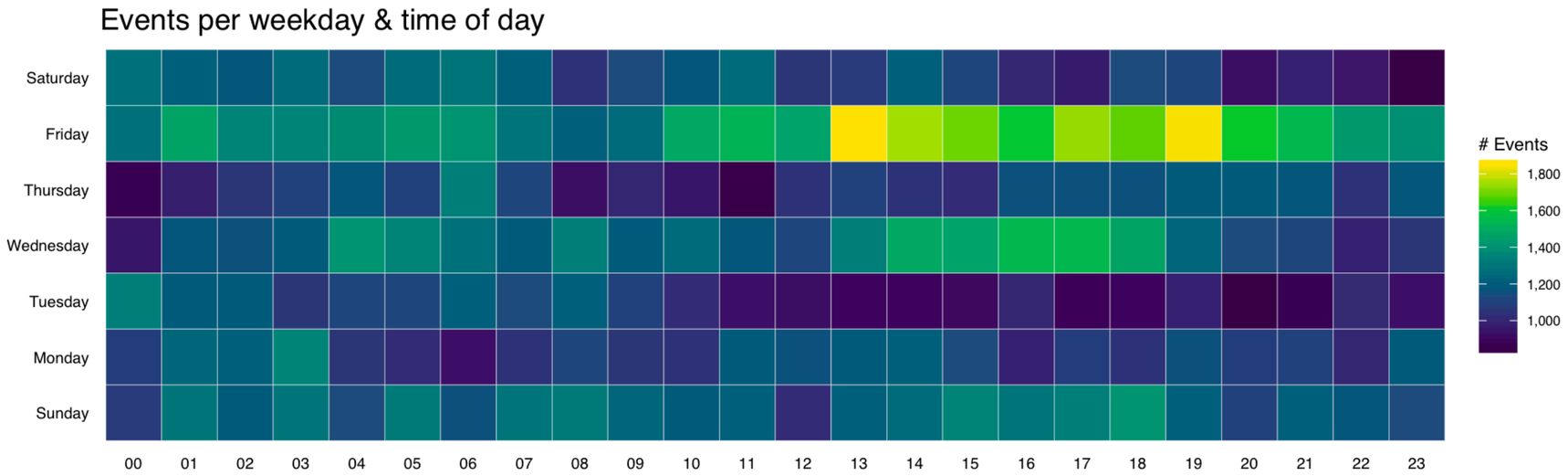
- **Many similarities to `lattice`:**
  - **automated legends and margins**
  - **easy to create panel plots***
  - **flexibility of `grid` system for manipulating graphics output**
  - **carefully chosen defaults**
- **BUT based on a grammar of graphics rather than a list of chart functions**

* also called trellis / lattice / small multiple / facet plots

So... why ggplot2?

- Modular system allows low level control with ease of a relatively high level system
- Intensionally extendable -- 199 packages on CRAN that begin with "gg"(!)
- Ability to create very professional, beautiful, publication ready plots
- Large, active community of users

Building block approach

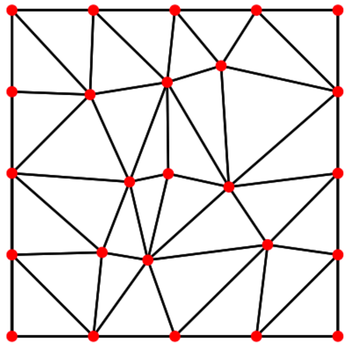


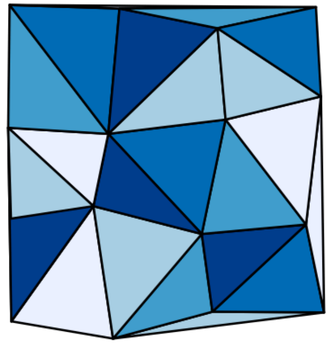
https://rud.is/b/2016/02/14/making-faceted-heatmaps-with-ggplot2/

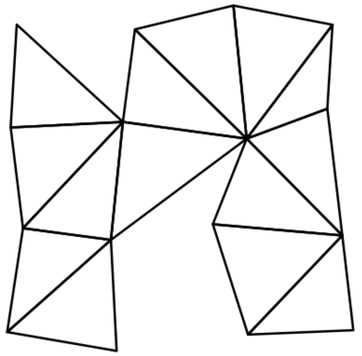
(I still use base R graphics)

- Graphs (histograms in particular) of vectors:
  `hist(x)`

- Graphics without real data

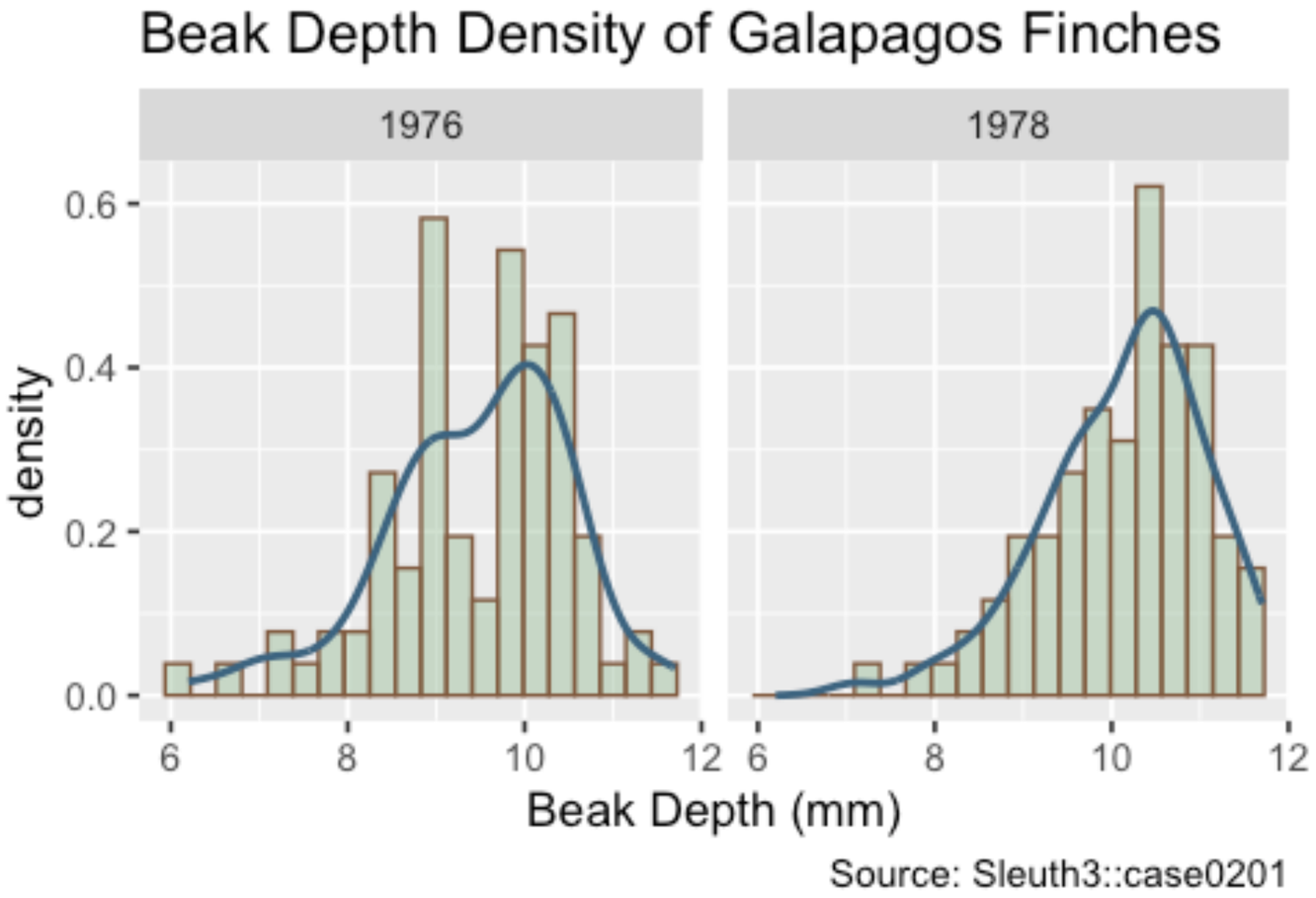






- When do you use base R graphics?

ggplot2 (quick example)

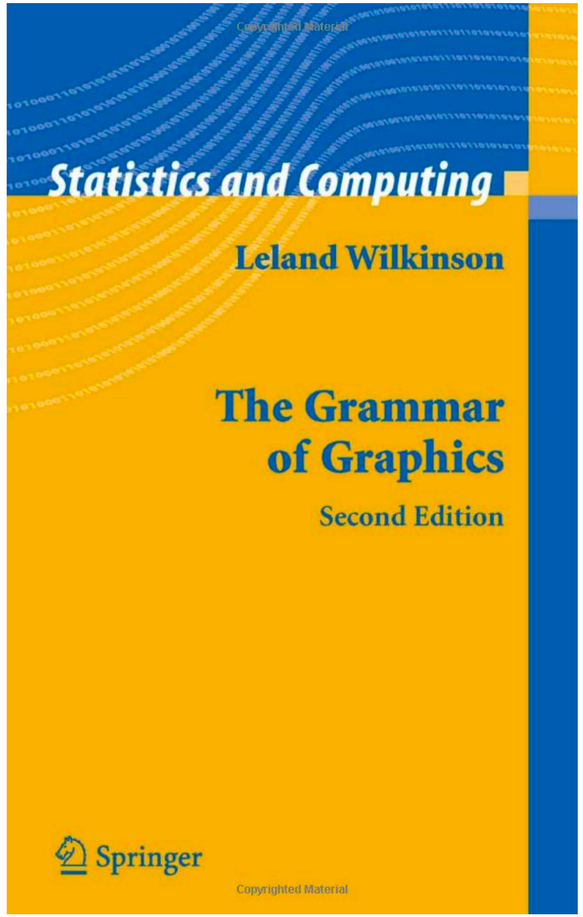


ggplot2 (quick example)

```
 1 library(ggplot2)
 2 finches <- Sleuth3::case0201
 3 ggplot(finches, aes(x = Depth, y = after_stat(density))) +
 4   geom_histogram(bins = 20, color = "#80593D",
 5                  fill = "#9FC29F", alpha = .5) +
 6   geom_density(color = "#3D6480", lwd = 1) +
 7   facet_wrap(~Year) +
 8   ggtitle("Beak Depth Density of Galapagos Finches") +
 9   labs(x = "Beak Depth (mm)",
10        caption = "Source: Sleuth3::case0201") +
11   theme_grey(13)
```

Grammar of Graphics

Leland Wilkinson



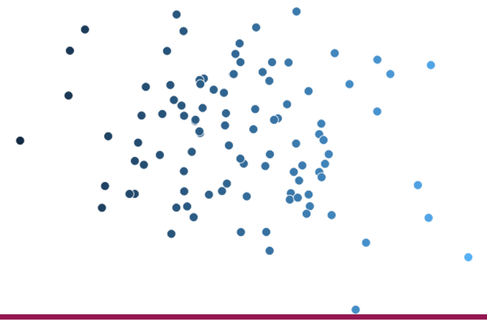
1944-2021

Grammar of graphics

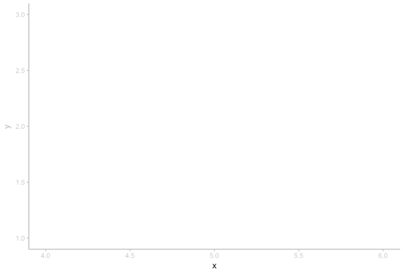
- presents a theory not a specific language / software
- takes us from "limited set of charts" to "an almost unlimited world of graphical forms"
- based on object oriented design: modular, reusuable
- other implementations exists besides ggplot2
- we will focus on the language/syntax of the ggplot2 implementation which differs slightly from the book

Building blocks

Layer(s)



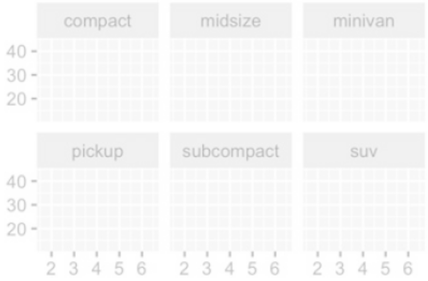
Scale(s)



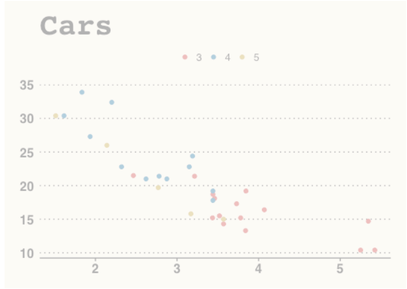
Coord

For now we will focus only on layers.

Facet

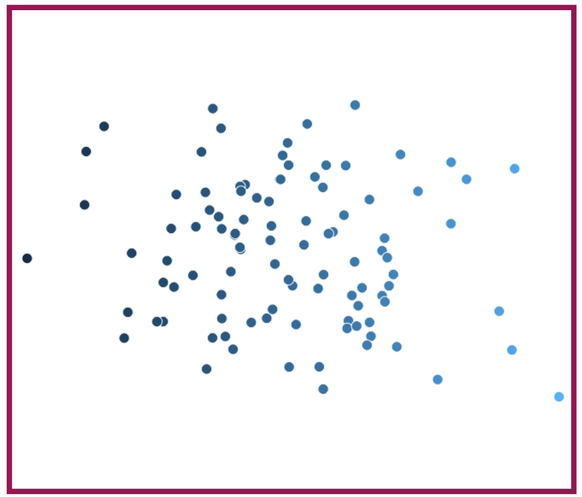


Theme



Layers

Each layer consists of:



1. GEOM
2. AESTHETIC MAPPING
3. DATA
4. STAT
5. POSITION

Layers

1. GEOM

point
bar
col
boxplot
line
histogram
density

geometric object

2. AESTHETIC MAPPING

x
y
color
fill
group
xmin
xmax
etc.

visual properties

→

variables

3. DATA

A	B	C

data frame

4. STAT

bin
boxplot
identity
density

statistical transformation

5. POSITION

identity
jitter
dodge
stack

shift

Layers

1. GEOM

point
bar
col
boxplot
line
histogram
density

2. AESTHETIC MAPPING

x
y
color
fill
group
xmin
xmax
etc.

3. DATA

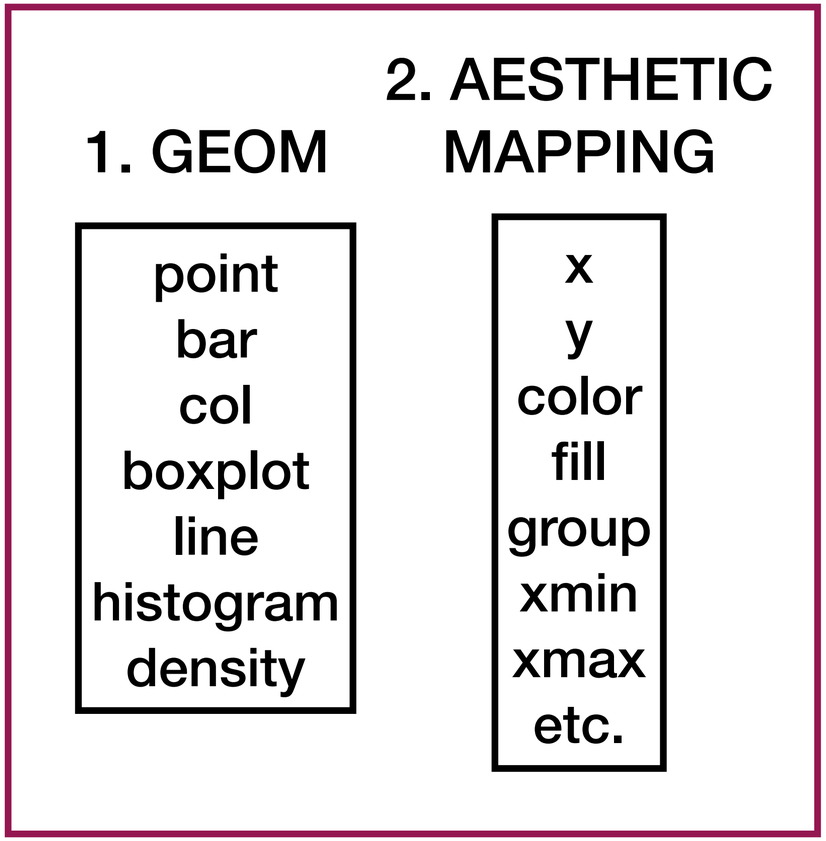
A	B	C

required

Most of the time you can use the default settings
for stat and position

GEOMs and mappings

- Think about a plot as a collection of GEOMs
- Each GEOM has required mappings
- For example `geom_histogram()` requires **x** (or **y**)
- Required mappings are sometimes indicated in bold in the help files (though not on the **posit** cheatsheet)



GEOMs and mappings

- Sometimes the mapping must be **continuous** or **discrete**, sometimes it can be either
- **continuous** = `numeric`
- **discrete** = `factor, character`
- Many mistakes are caused by data in the wrong form, for example, numeric classified as character data