



BUSINESS
SCHOOL

SCIENCE POLICY
RESEARCH UNIT

Data visualisation

Daniele Rotolo

Introductory Data Science for Innovation (995N1) – Weeks 3, 11 October 2021

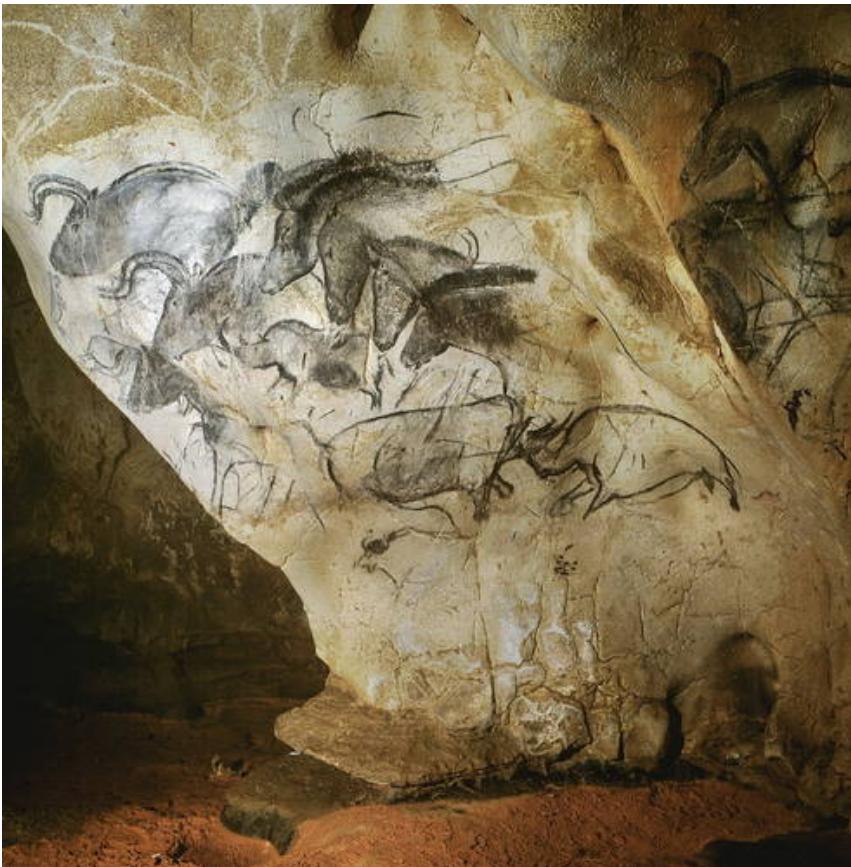
Outline

Outline

- Why visualising data?
- Visualisation types
- Beyond statistics and visualisation
- Data visualisation in R

Why visualising data?

Why visualising data?



- Half of our brain is dedicated to processing **visual signals**
- We can process images **60,000 times faster** than text

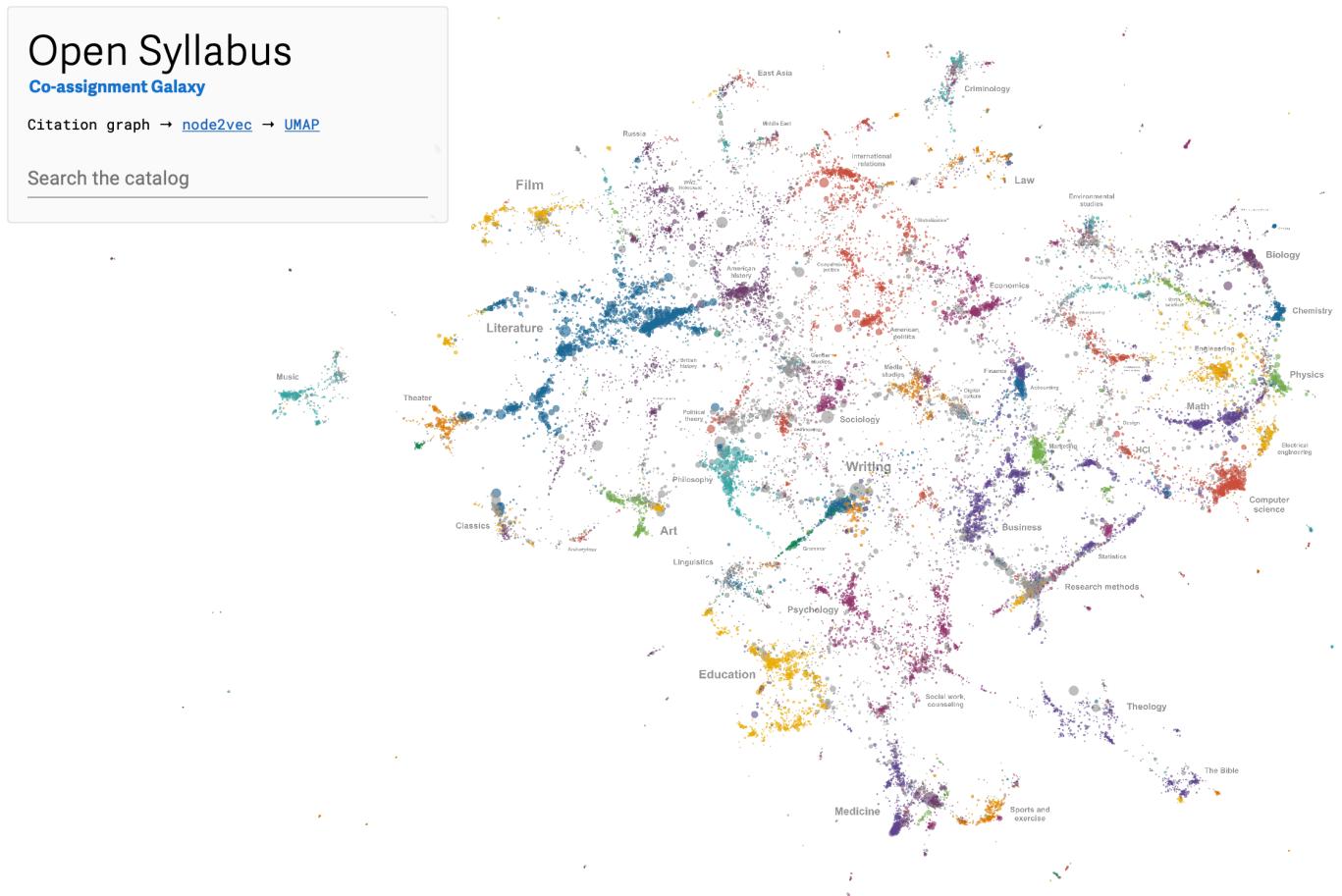
Source: Figurative drawings of the Grotte Chauvet-Pont d'Arc, 30,000-32,000 BP

(<http://whc.unesco.org/>)

Why visualising data?

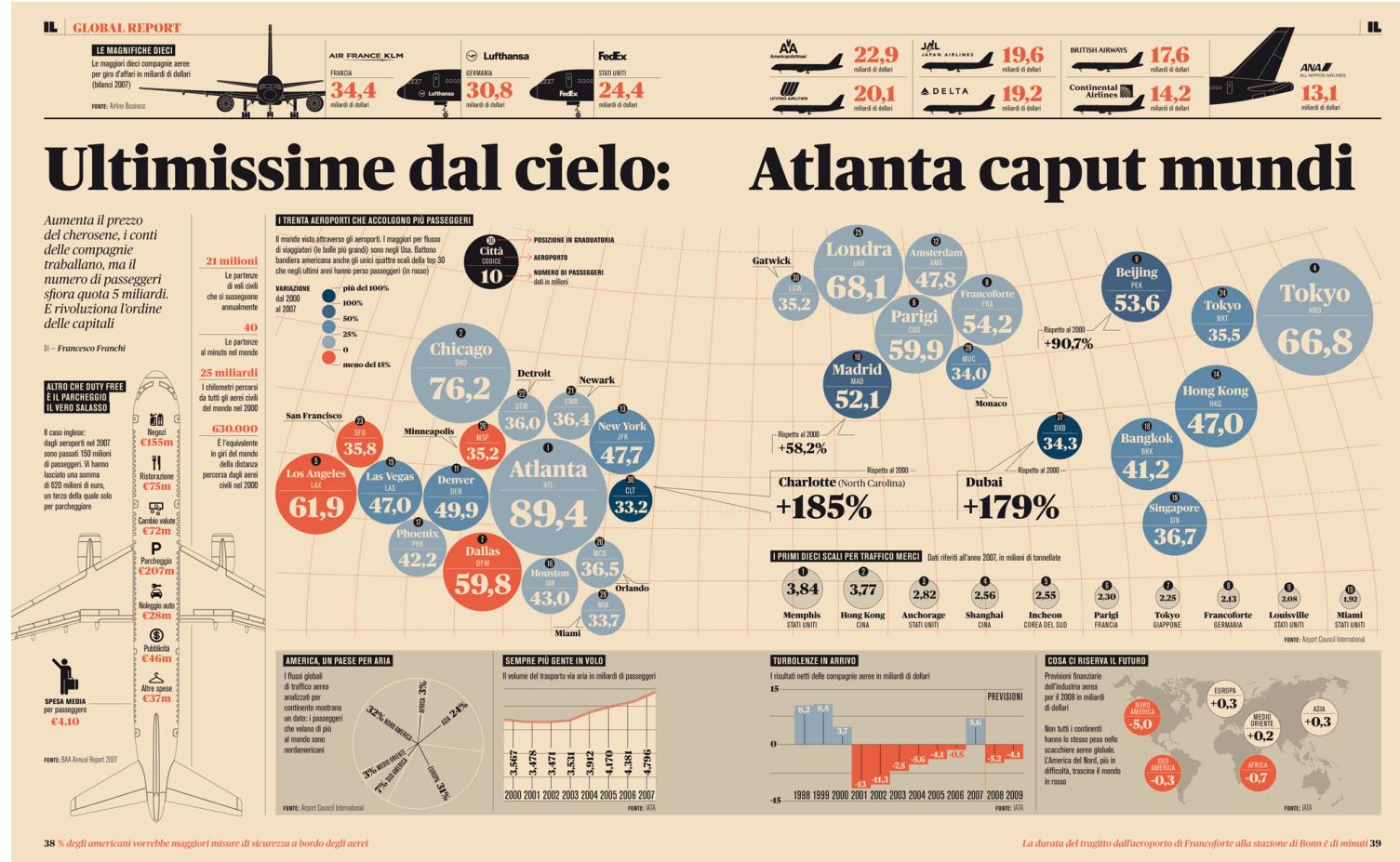
- Increasing capabilities to collect, store and analyse data
 - government data
 - APIs
 - ...
- Large amount of real time data
- Risk of information overload
- Data visualisation
 - to simplify the complexity
 - to identify actual and misleading patterns

... to simplify the complexity



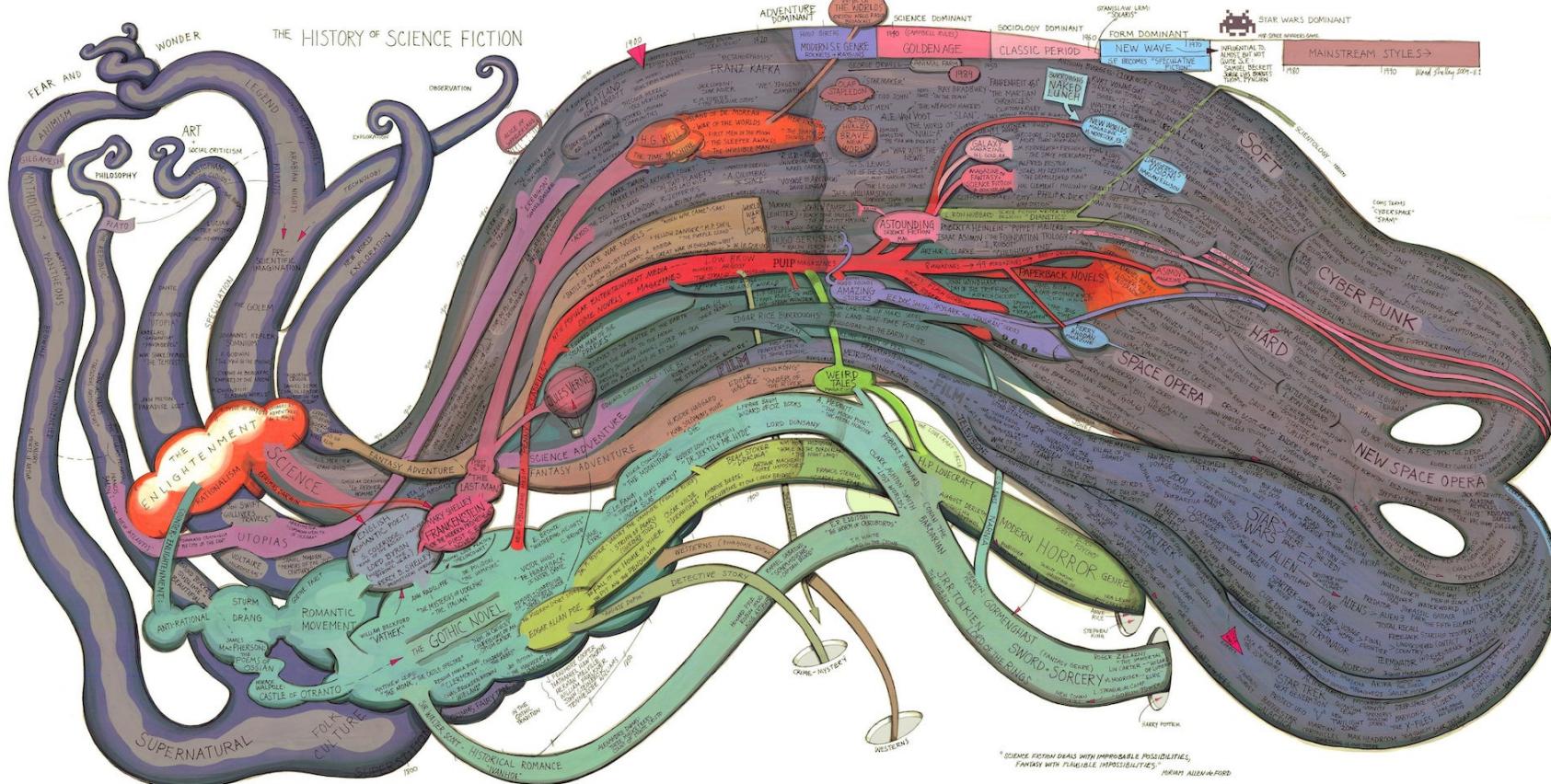
Source: <https://opensyllabus.org>

... to simplify the complexity



Source: Il Sole 24 Ore, November 2008

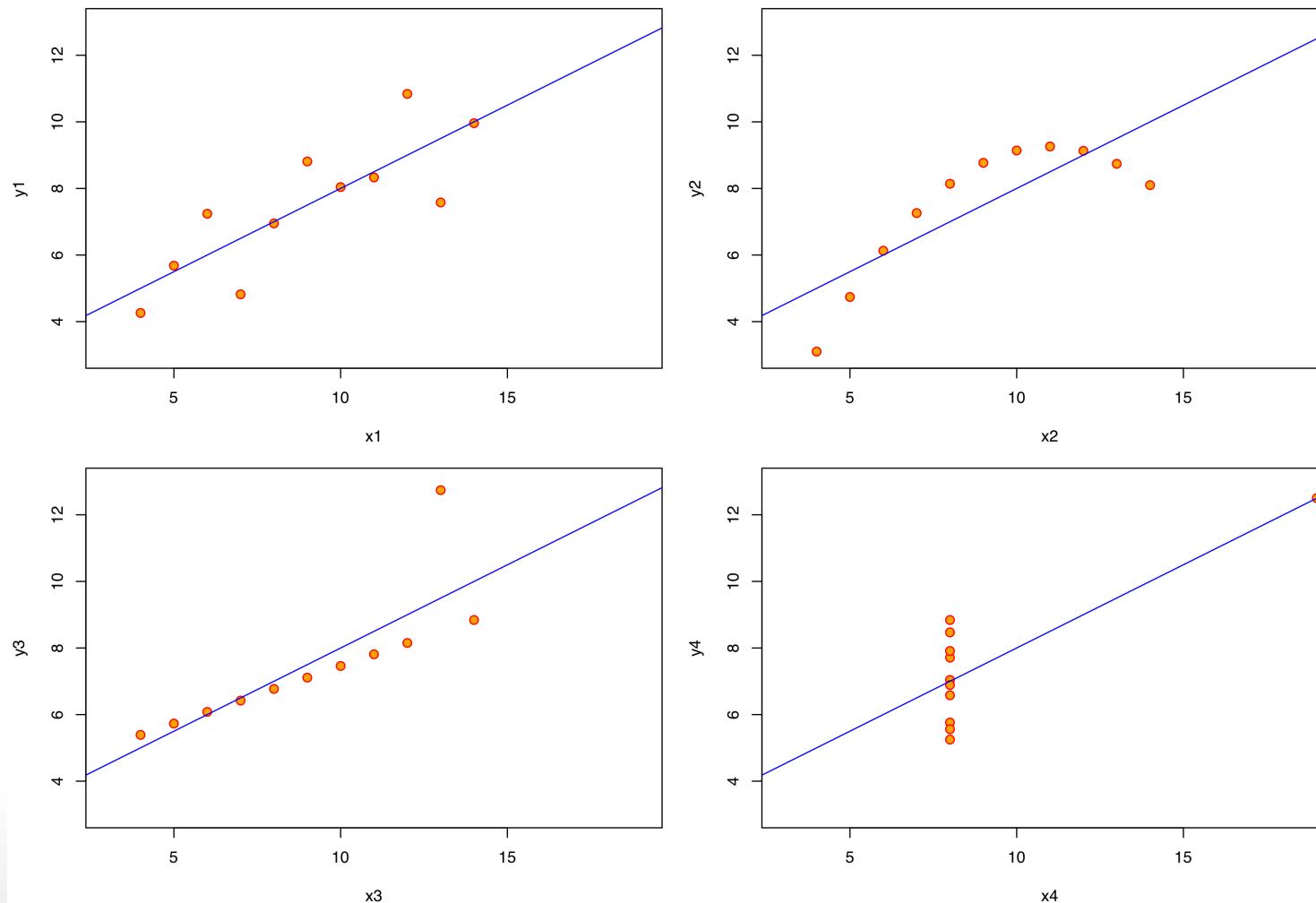
... to simplify the complexity



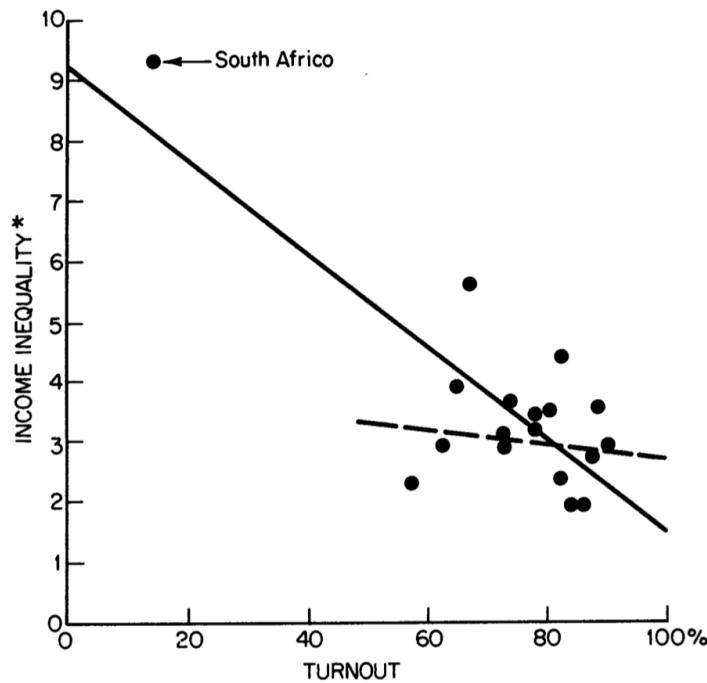
Source: VanderMeer (2013)

... to identify actual and misleading patterns

Anscombe's 4 Regression data sets



... to identify actual and misleading patterns



Key. — bivariate slope including South Africa (N=18)
- - - bivariate slope excluding South Africa (N=17)

*Income inequality is defined as the ratio of income received by the wealthiest population quintile to that received by the poorest 40 percent of the population.

Source: Jackman (1980)

... to identify actual and misleading patterns

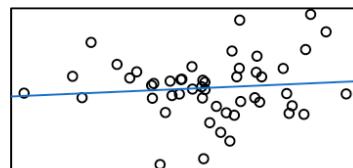
- [Jan Vanhove](#) has provided a script in R to explore the diversity of patterns that may lie behind a simple correlation measure
- You can specify the correlation (`r`) and the number of observations (`n`)

```
source("http://janhove.github.io/RCode/plot_r.R")
plot_r(r = 0.1, n = 50)
```

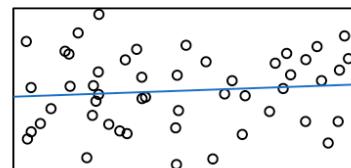
... to identify actual and misleading patterns

All correlations: $r(50) = 0.1$

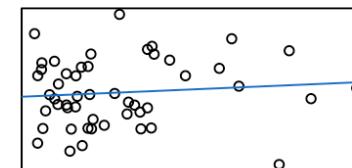
(1) Normal x, normal residuals



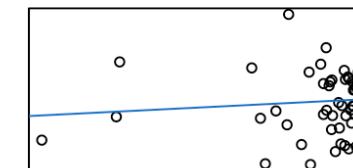
(2) Uniform x, normal residuals



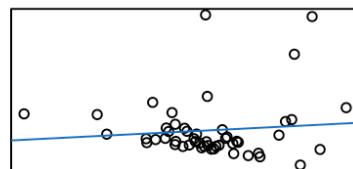
(3) +-skewed x, normal residuals



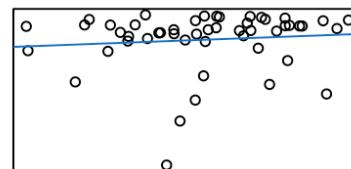
(4) --skewed x, normal residuals



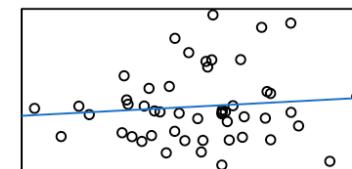
(5) Normal x, +-skewed residuals



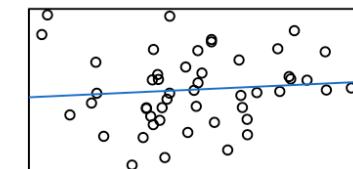
(6) Normal x, --skewed residuals



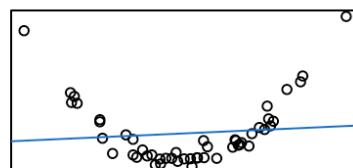
(7) Increasing spread



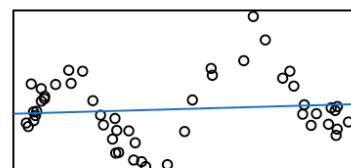
(8) Decreasing spread



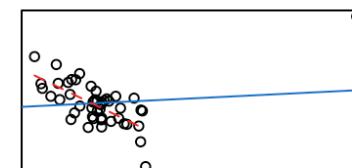
(9) Quadratic trend



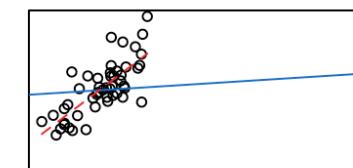
(10) Sinusoid relationship



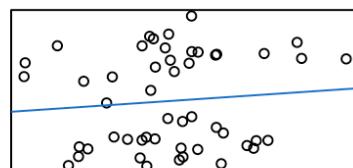
(11) A single positive outlier



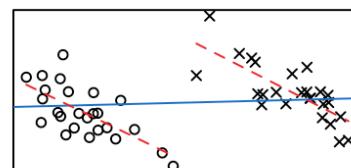
(12) A single negative outlier



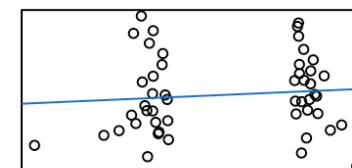
(13) Bimodal residuals



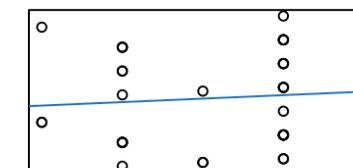
(14) Two groups



(15) Sampling at the extremes



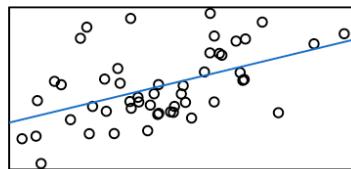
(16) Coarse data



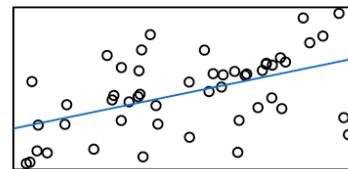
... to identify actual and misleading patterns

All correlations: $r(50) = 0.5$

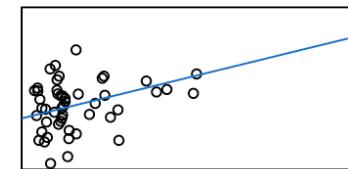
(1) Normal x, normal residuals



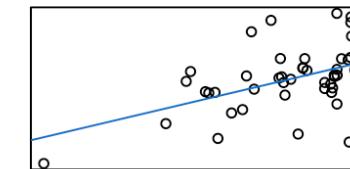
(2) Uniform x, normal residuals



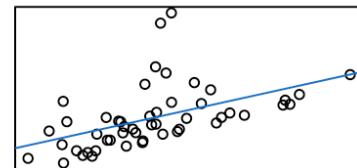
(3) +-skewed x, normal residuals



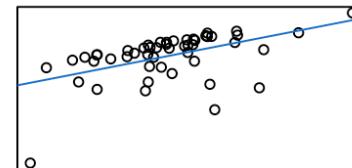
(4) --skewed x, normal residuals



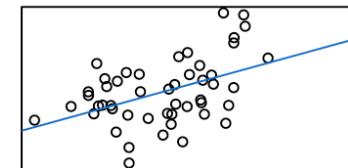
(5) Normal x, +-skewed residuals



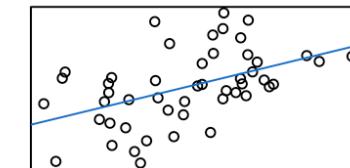
(6) Normal x, --skewed residuals



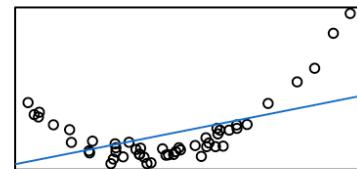
(7) Increasing spread



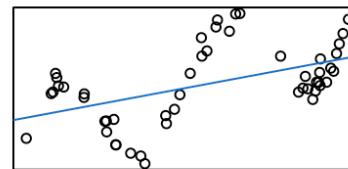
(8) Decreasing spread



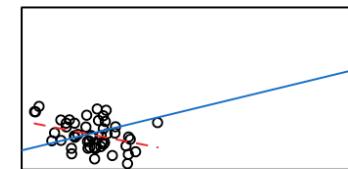
(9) Quadratic trend



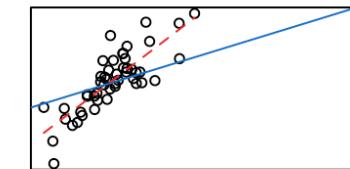
(10) Sinusoid relationship



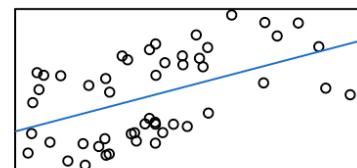
(11) A single positive outlier



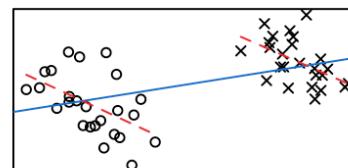
(12) A single negative outlier



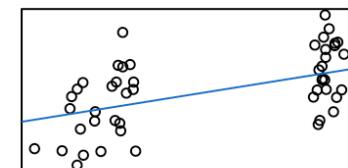
(13) Bimodal residuals



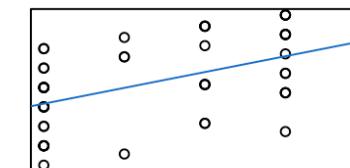
(14) Two groups



(15) Sampling at the extremes



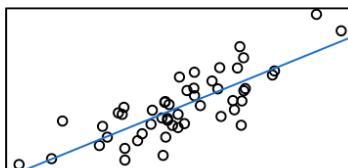
(16) Coarse data



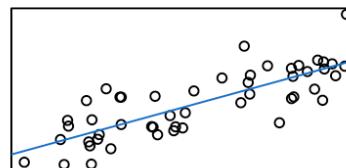
... to identify actual and misleading patterns

All correlations: $r(50) = 0.8$

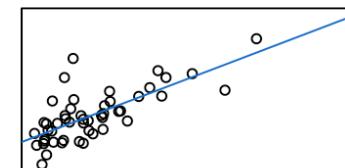
(1) Normal x, normal residuals



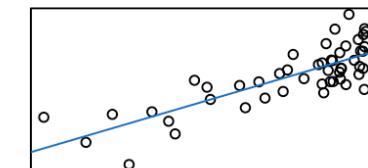
(2) Uniform x, normal residuals



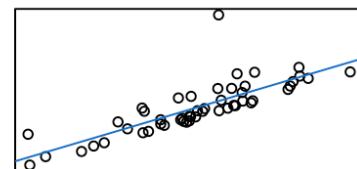
(3) +-skewed x, normal residuals



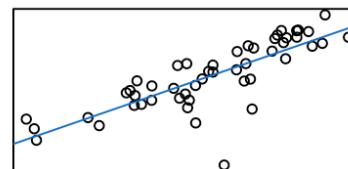
(4) --skewed x, normal residuals



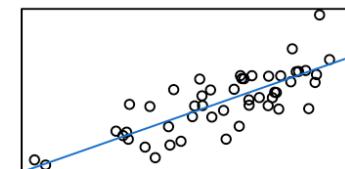
(5) Normal x, +-skewed residuals



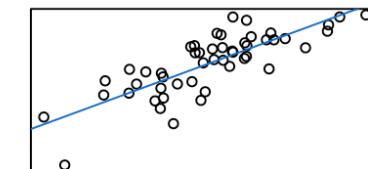
(6) Normal x, --skewed residuals



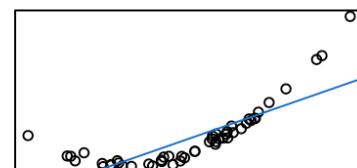
(7) Increasing spread



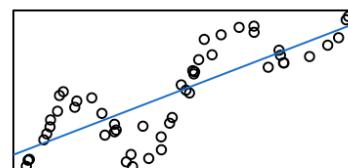
(8) Decreasing spread



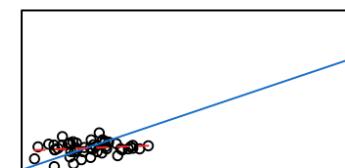
(9) Quadratic trend



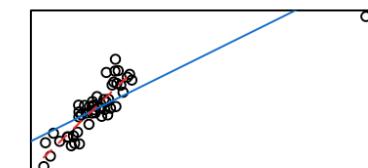
(10) Sinusoid relationship



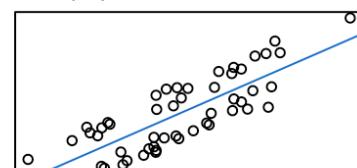
(11) A single positive outlier



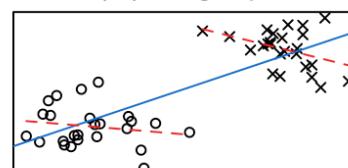
(12) A single negative outlier



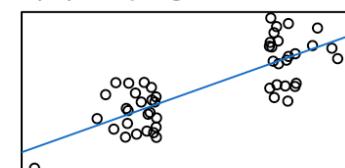
(13) Bimodal residuals



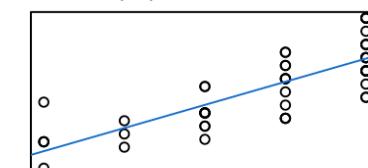
(14) Two groups



(15) Sampling at the extremes



(16) Coarse data



Visualisation types

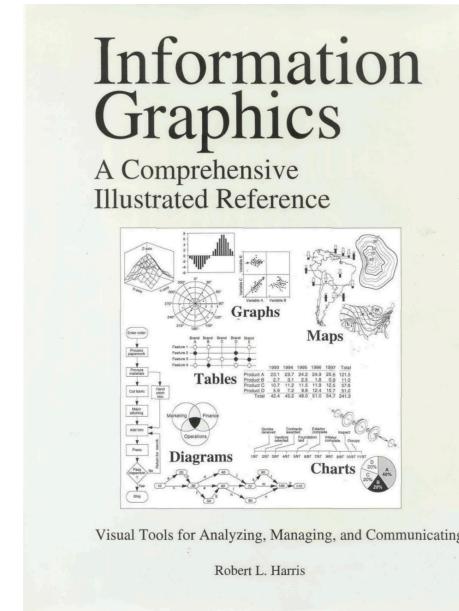
Visualisation types

Visualisation types by Börner (2016)

- **Tables:** values arranged in rows and columns (e.g. summary tables)
- **Charts:** qualitative and quantitative representations of data without a reference system (e.g. pie charts)
- **Graphs:** qualitative and quantitative representations of data with a reference system (e.g. scatterplot)
- **Maps:** geographical distribution of data (e.g. geographical maps)
- **Networks layouts:** data represented as sets of nodes and links between these, i.e. relationships between nodes (e.g. co-occurrence networks)

Visualisation types

- One type of visualisation can be **converted** into another (e.g. a pie chart can be transformed into a bar chart)
- Different visualisation types can be **combined** to examine data from different a number of angles
- See Harris (1996) for a comprehensive list of **visualisation types**



Tables: *One way*

Name	Age
Alice	6
Bill	7
Harry	4
Sue	5

Two variables along the vertical axis without rules

Name	Age
Alice	6
Bill	7
Harry	4
Sue	5

Two variables along vertical axis with rules

Two variables along horizontal axis with rules

Name	Alice	Bill	Harry	Sue
Age	6	7	4	5

Three variations of one-way tables

Source: Harris (1996)

Tables: Two way

The diagram illustrates two ways of presenting the same data in a two-way table. The top part shows a table with Product categories on the vertical axis and years on the horizontal axis. The bottom part shows the same data rotated, with years on the vertical axis and products on the horizontal axis. Arrows indicate the variables: one variable along the vertical axis, second variable along the horizontal axis, and a third variable in the body of the table (shaded area).

	1990	1991	1992
Product A	2	4	6
Product B	10	12	17
Product C	8	7	6
Product D	22	29	20

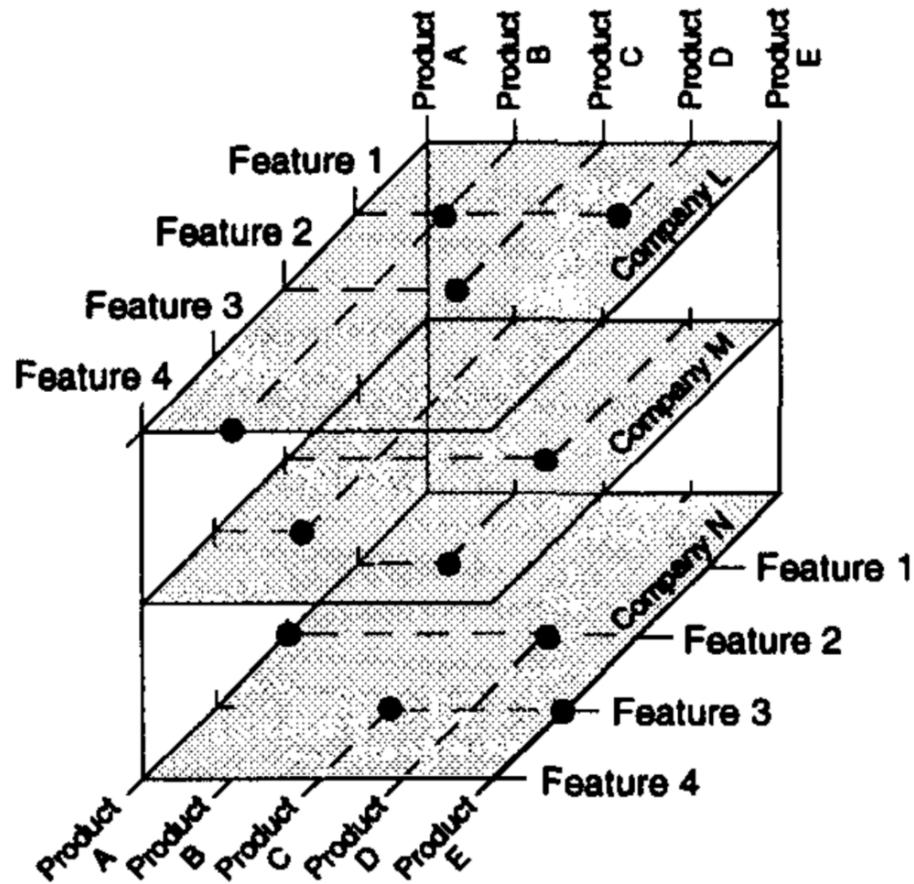
	Product A	Product B	Product C	Product D
1990	2	10	8	22
1991	4	12	7	29
1992	6	17	6	20

One variable along vertical axis Second variable along horizontal axis Third variable in body of table (shaded area)

Two variations of two-way tables using the same data

Source: Harris (1996)

Tables: 3D



Source: Harris (1996)

Tables: ...

... many other variations

Tables: Terminology

Terminology and location of key elements

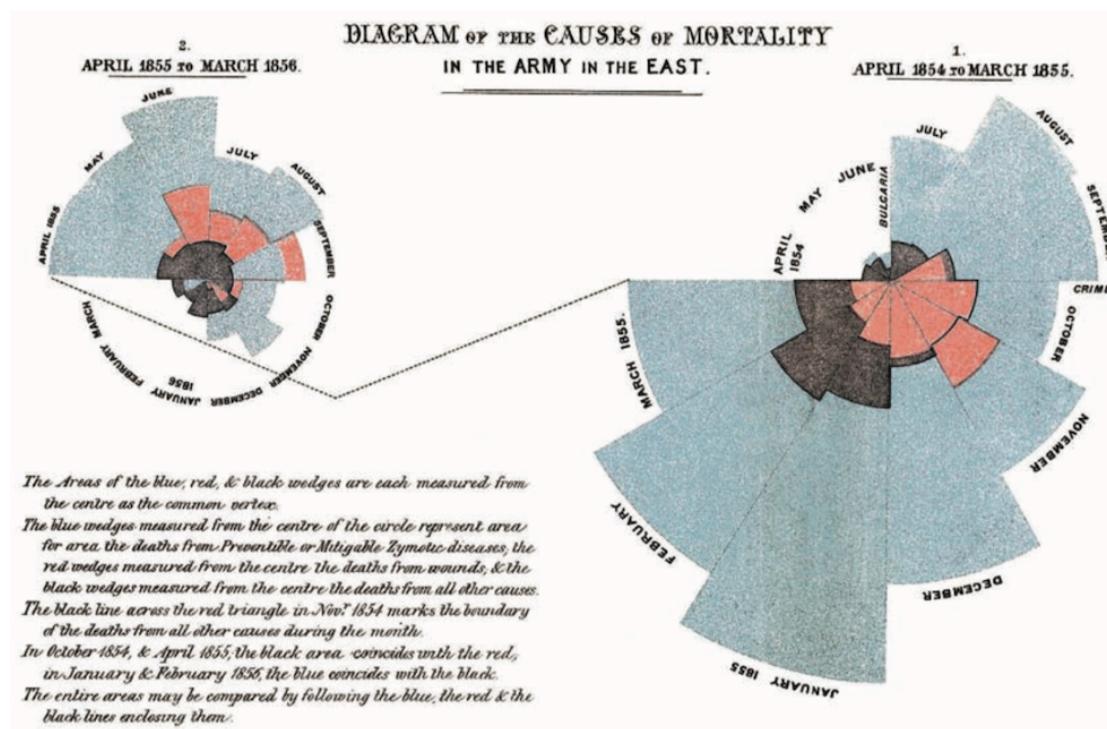
Sales Predictions for 1996 Planning Meeting (By Product Line)					
Product	Sales				
	1995 Estimate		1999 Forecast		
	Dollars (000)	% of total	Dollars (000)	% of total	
Drills	Large	14.4	8.6	25.6	12.2
	Medium	28.2	16.9	45.6	21.7
	Small	40.8	24.5	76.4	36.4
Nail Drivers	Large	8.7	5.4	13.2	6.3
	Small	11.4	6.8	15.1	7.2
Torque Wrench	Large	6.2	3.7	6.1	2.8
	Small	8.9	5.3	11.3	5.4
	Pneumatic	4.3	2.6	7.1	3.4
	Electric	5.9	3.5	5.0	2.4
Saws		37.9	22.7	4.6	2.2
	Total	166.7	100.0	210.0	100.0

If multiple terms are used to describe the same thing, each of the terms are shown above. • When tables are used in specific applications, parts of the tables are sometimes given additional names related specifically to the application. For example, when used in a database, a column might be called a field and a row referred to as a record.

Source: Harris (1996)

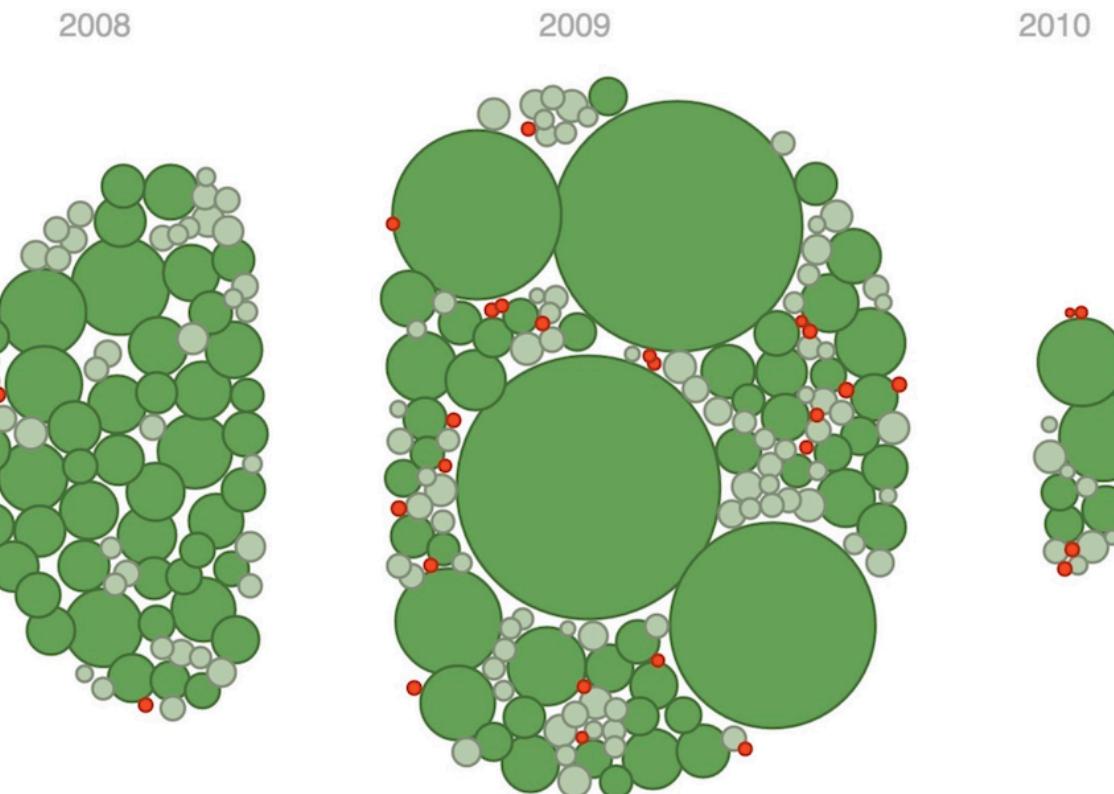
Charts: *Pie charts*

Causes of Mortality in the British Military during the Crimean War



Source: Börner (2016)

Charts: *Bubble charts*



Source: [Gates Foundation's education-based donations](#) (color and size represent grant amount)

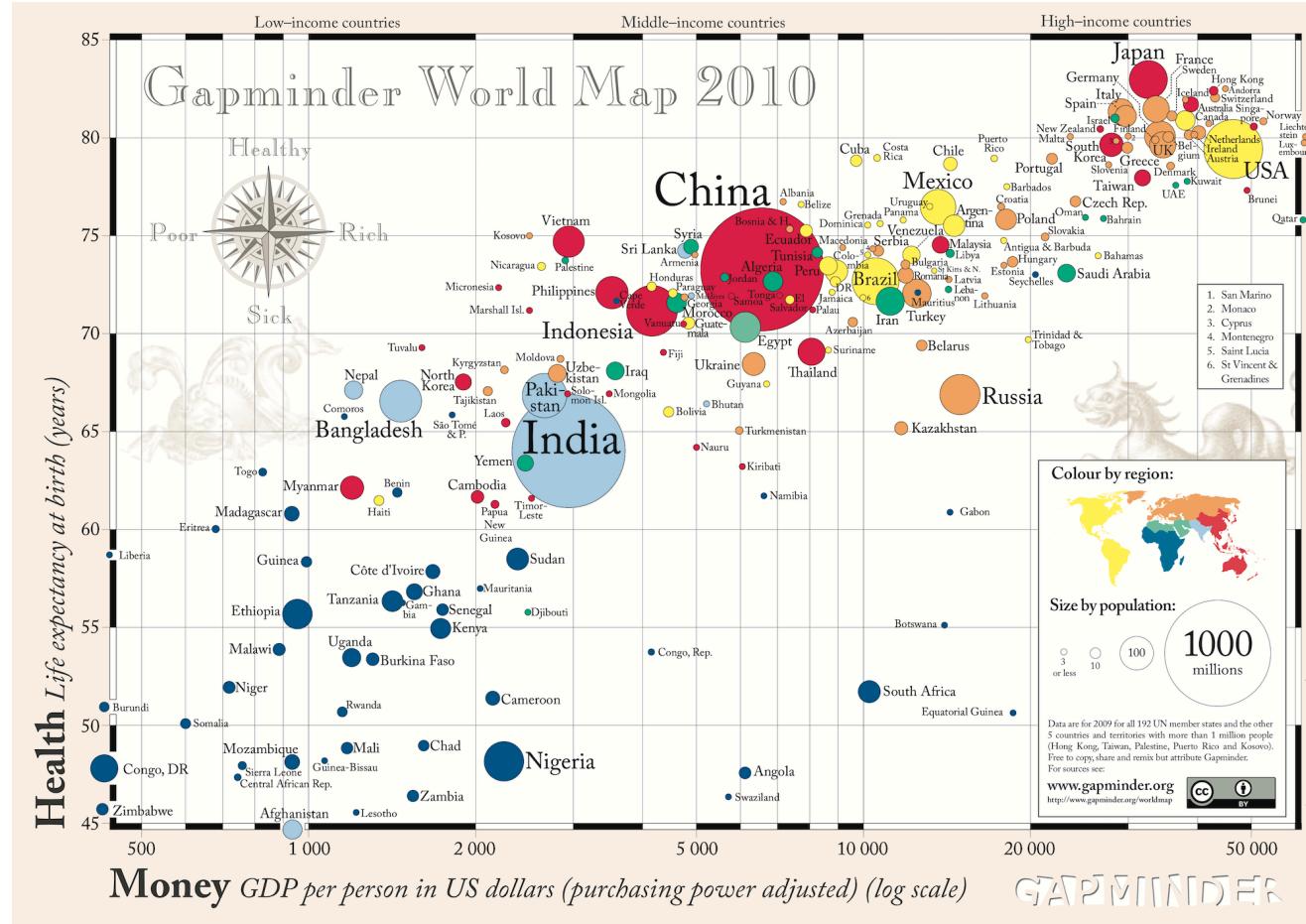
Charts: *Word clouds*



Charts: ...

... many other variations

Graphs: *Bubble graphs*

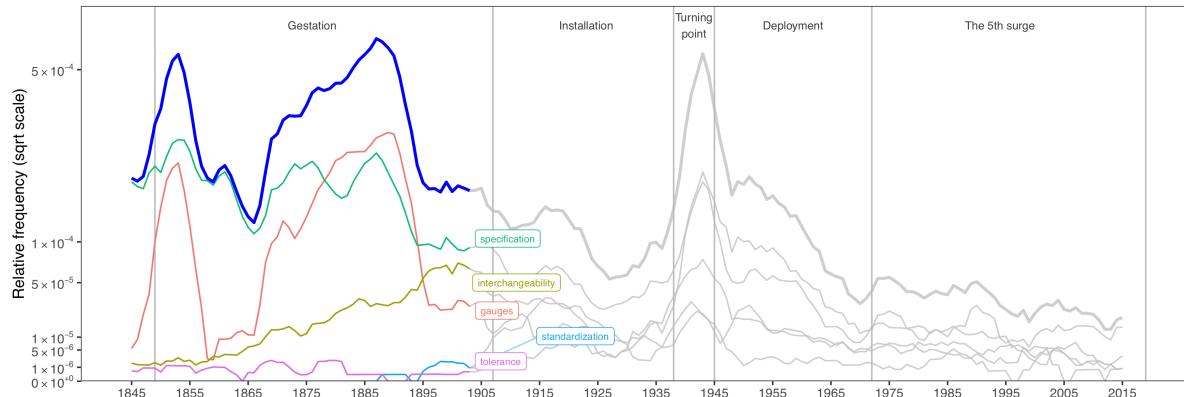


Source: <https://www.gapminder.org>

Graphs: *Line graphs*

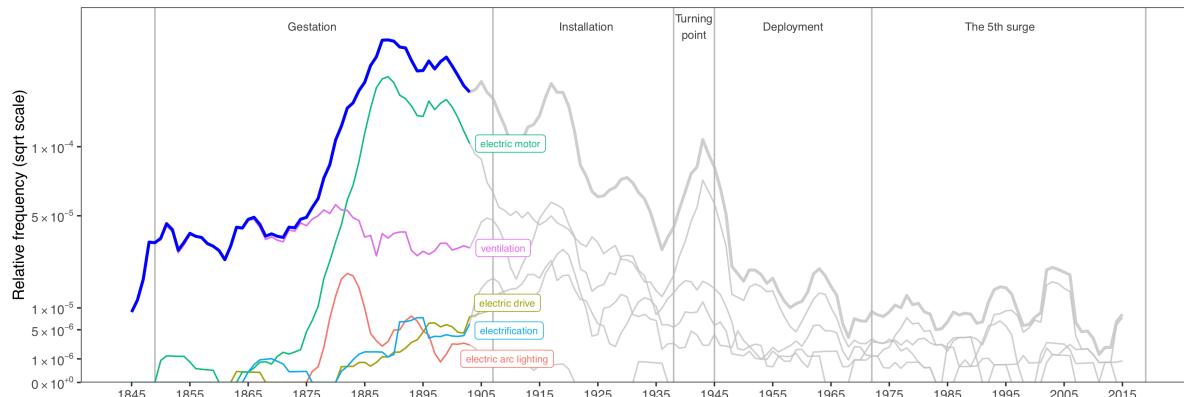
A

Rule: Make parts interchangeable (blue line represents the aggregated trend)



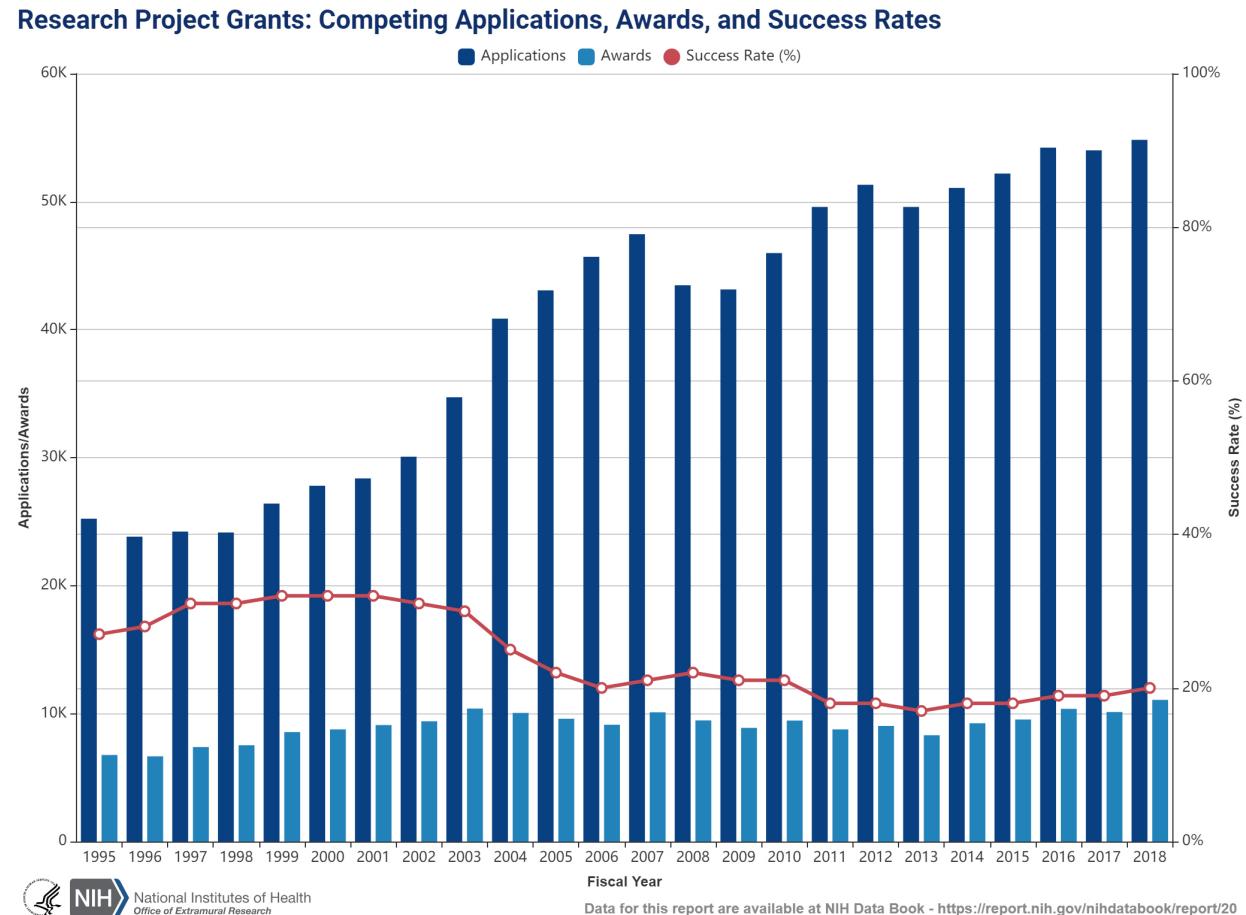
B

Rule: Electrify factories, work environments and machines (blue line represents the aggregated trend)



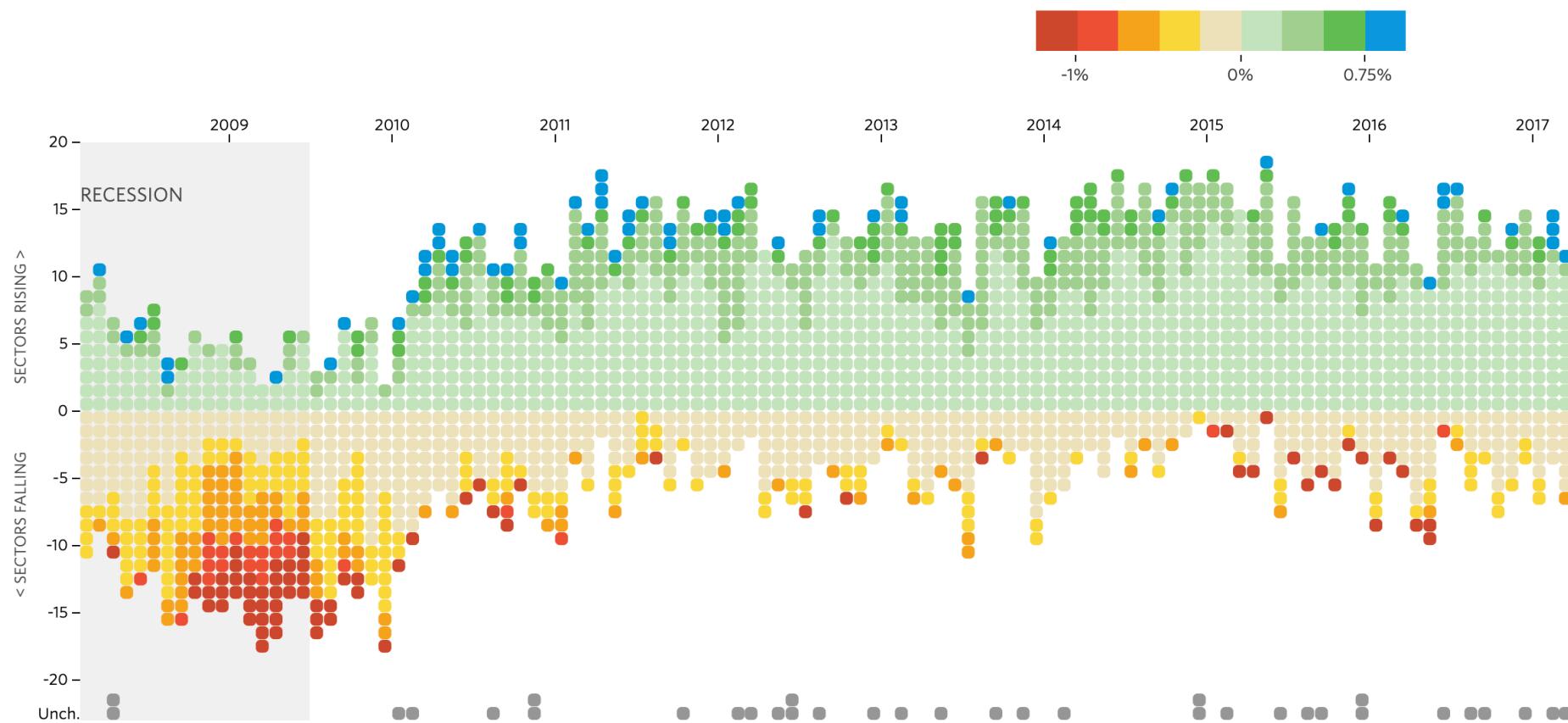
Source: Bone and Rotolo (2020)

Graphs: *Line and bar graphs*



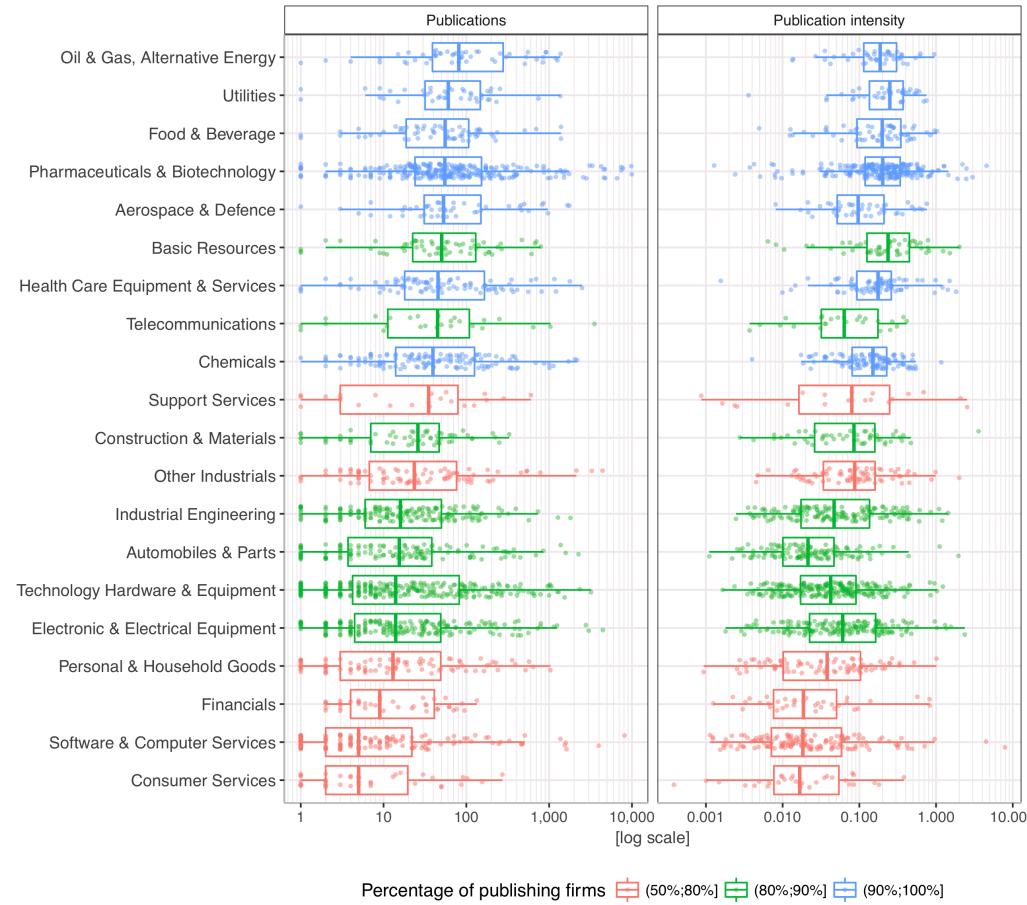
Source: <https://www.nih.gov/news-events/nih-research-grants-digital-press-kit>

Graphs: *Interactive bar graphs*



Source: Winners and Losers: Job Gains and Losses, [Wall Street Journal](#)

Graphs: Box plot



Source: Camerani, Rotolo, and Grassano (2018)

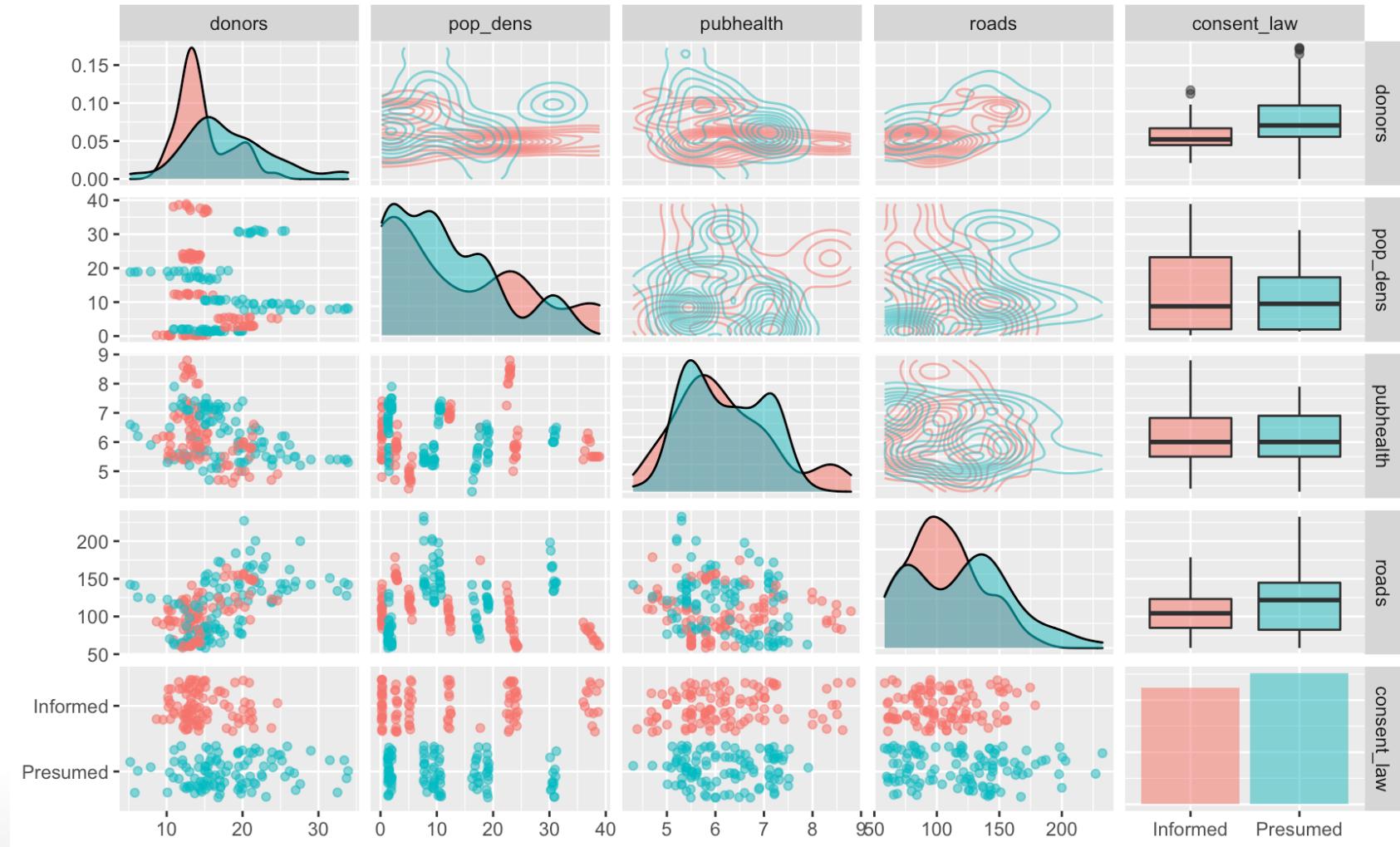
Graphs: *Statistical visualisations*

```
library(tidyverse)
library(GGally)
load(file = "organdata.rda")

organdata_sm <- organdata %>%
  select(donors, pop_dens, pubhealth,
         roads, consent_law)

g <- ggpairs(data = organdata_sm,
              mapping = aes(color = consent_law, alpha = 0.5),
              upper = list(continuous = wrap("density"), combo = "box_no_facet"),
              lower = list(continuous = wrap("points"), combo = wrap("dot_no_facet")))
```

Graphs: Statistical visualisations



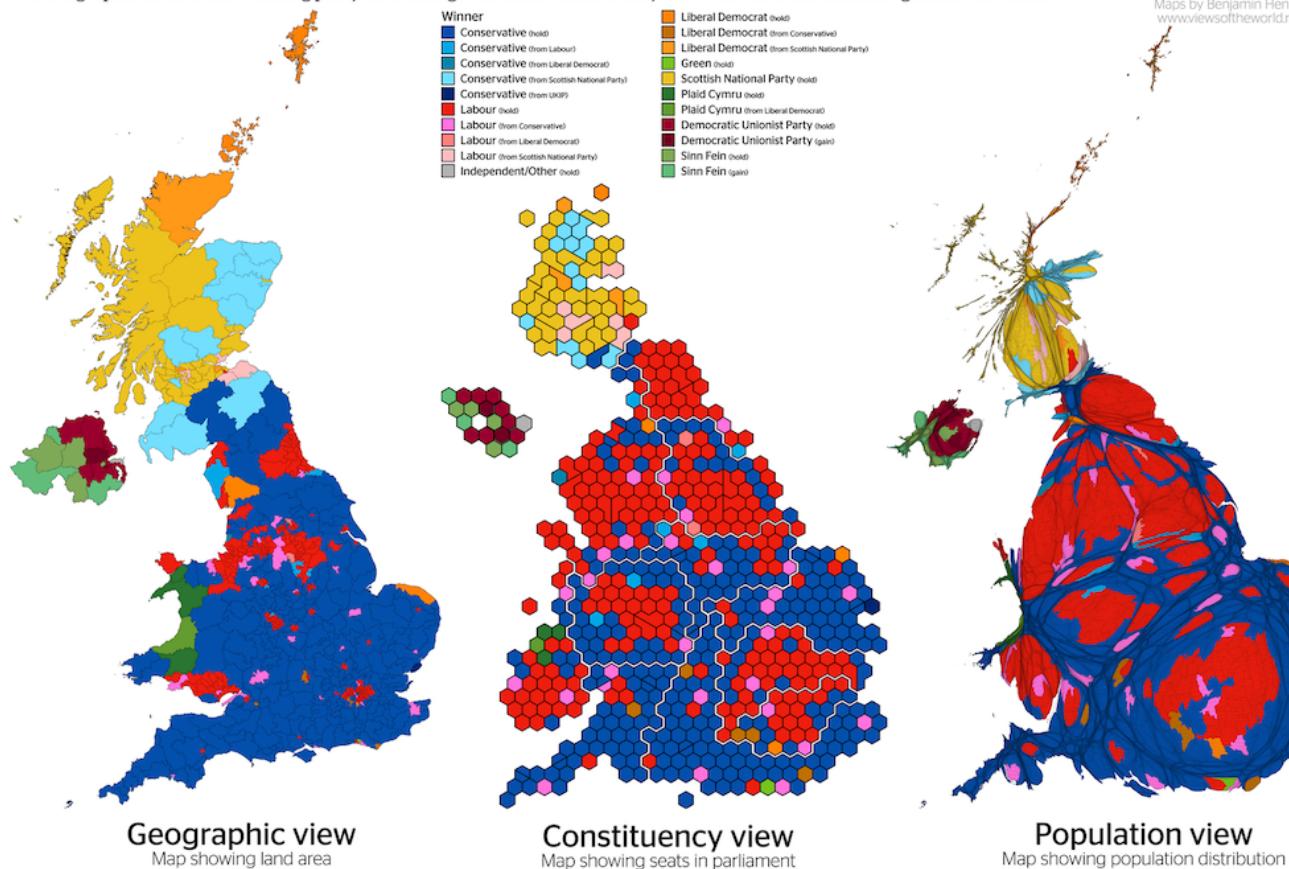
Graphs: ...

... many other variations

Maps: *Cartograms*

Views of the 2017 UK General Election

A cartographic look at the winning party and changes in each constituency between the 2015 and 2017 general elections

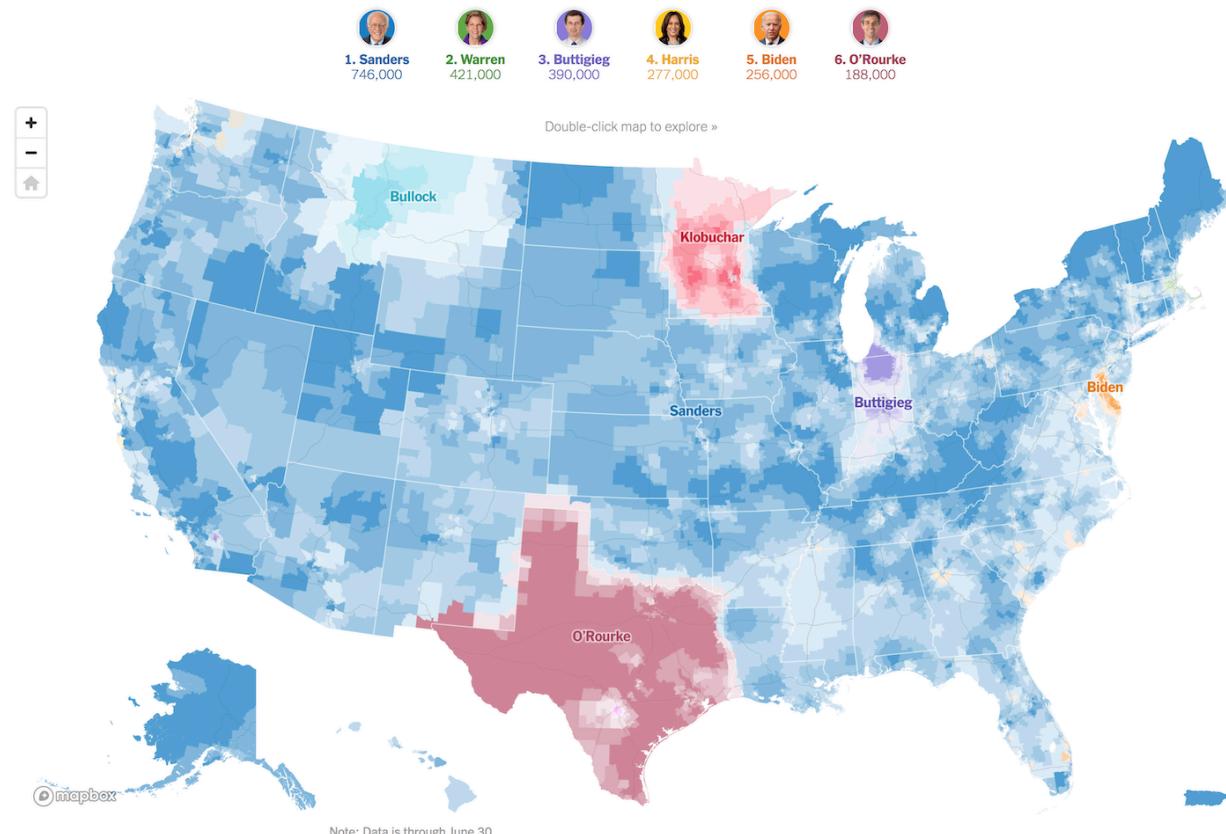


WORLD
MAPPER
Maps by Benjamin Hennig
www.viewsoftheworld.net

Cartographic analysis created for <http://www.selectionanalysis.co.uk> & <http://geographical.co.uk>

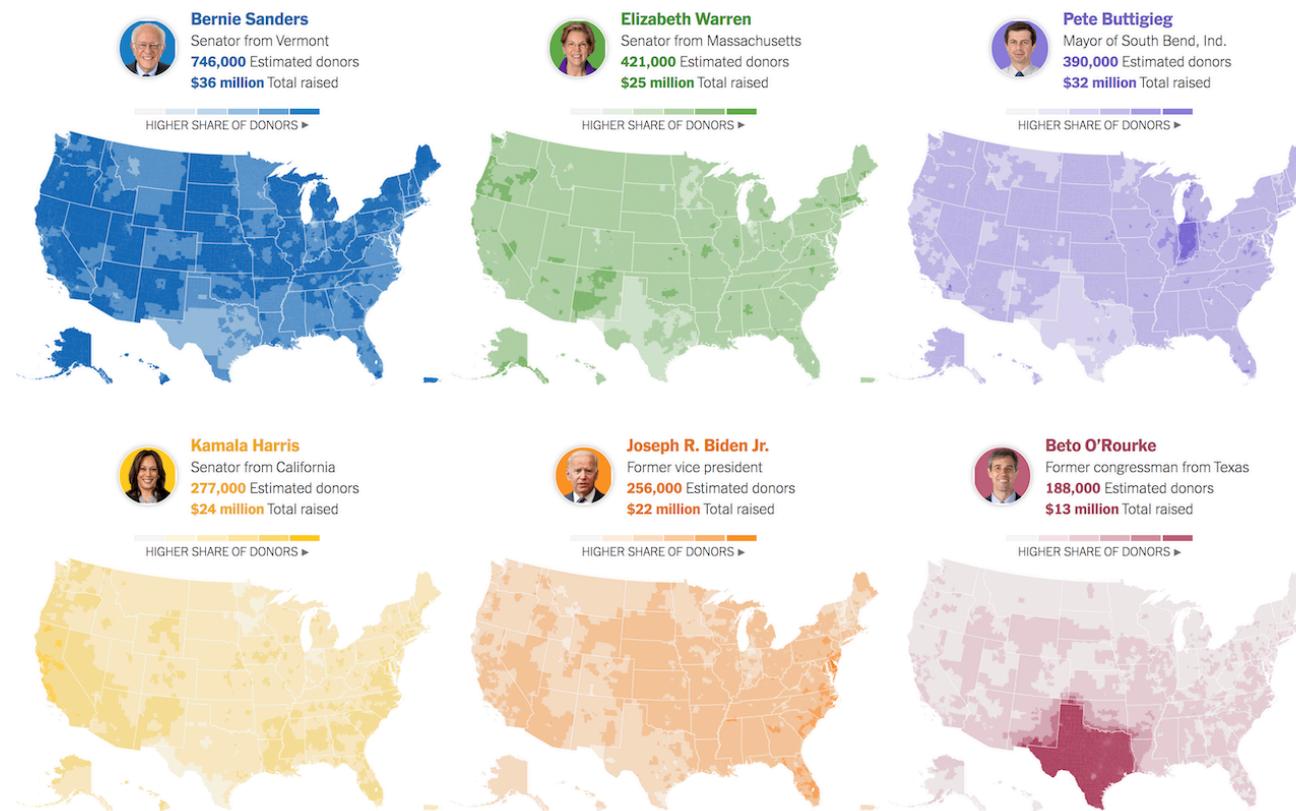
Source: <https://geographical.co.uk/places/mapping/item/2276-mapping-the-2017-general-election>

Maps: *Choropleth maps*



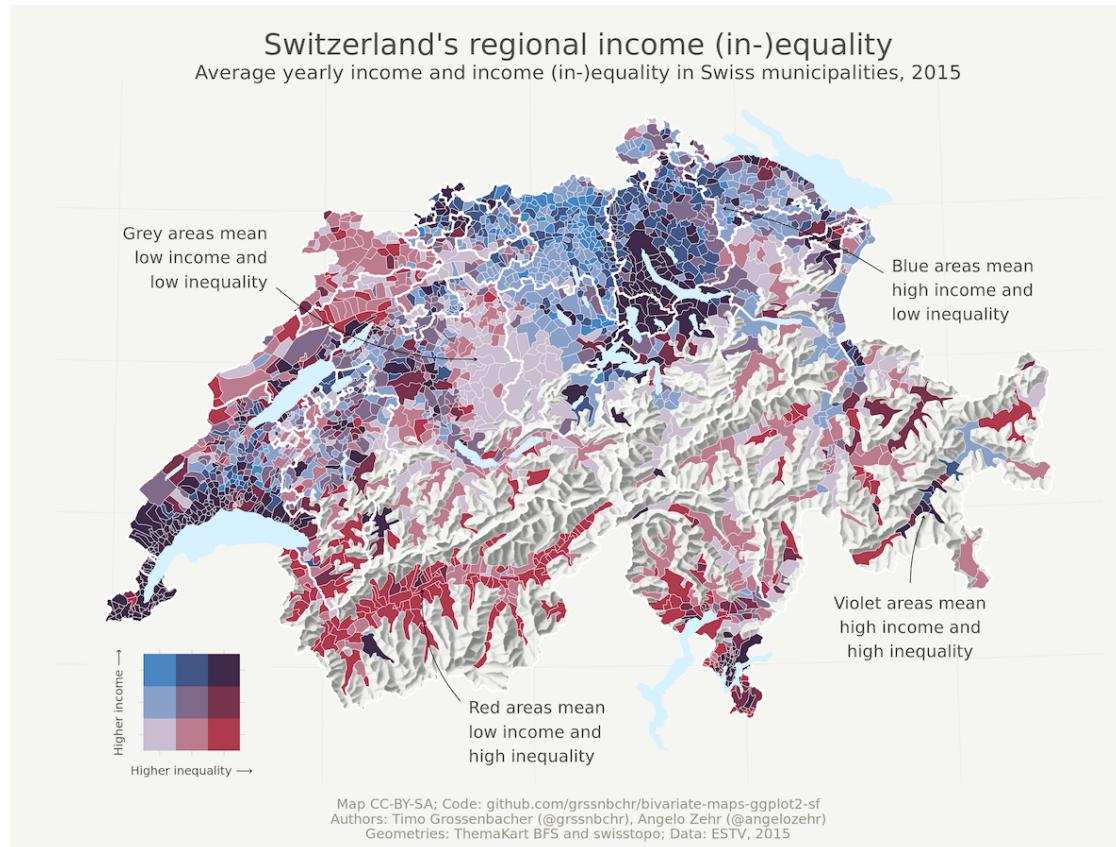
Source: <https://www.nytimes.com/interactive/2019/08/02/us/politics/2020-democratic-fundraising.html>

Maps: *Choropleth maps*



Source: <https://www.nytimes.com/interactive/2019/08/02/us/politics/2020-democratic-fundraising.html>

Maps: *Choropleth maps*



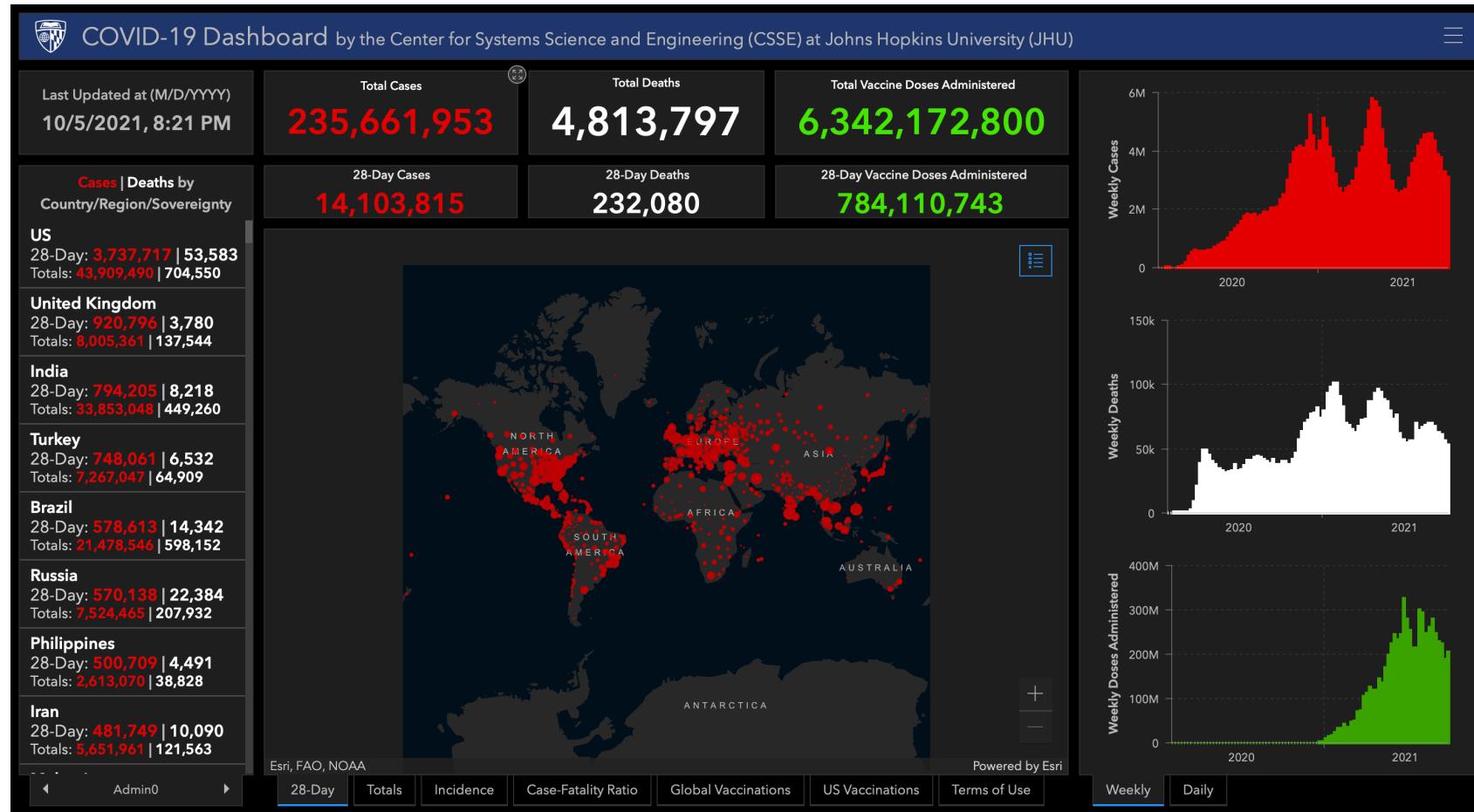
Source: <https://timogrossenbacher.ch/2019/04/bivariate-maps-with-ggplot2-and-sf/>

Maps: *Choropleth maps*



Source: <https://centerforcollectivelearning.org/urbanperception>

Maps: Proportional symbol maps

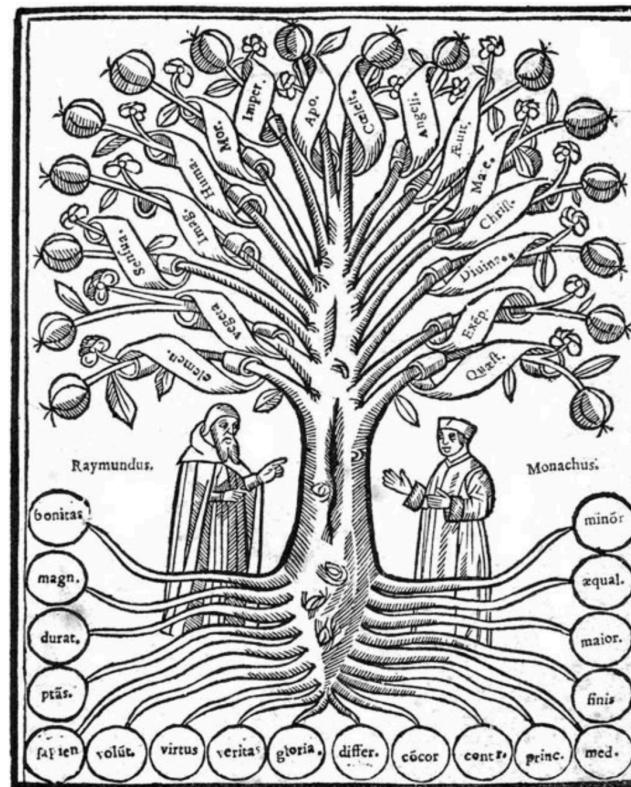


Source: Active COVID-19 cases on 5 October 2021 (<https://www.gisaid.org/epiflu-applications/global-cases-covid-19/>)

Maps: ...

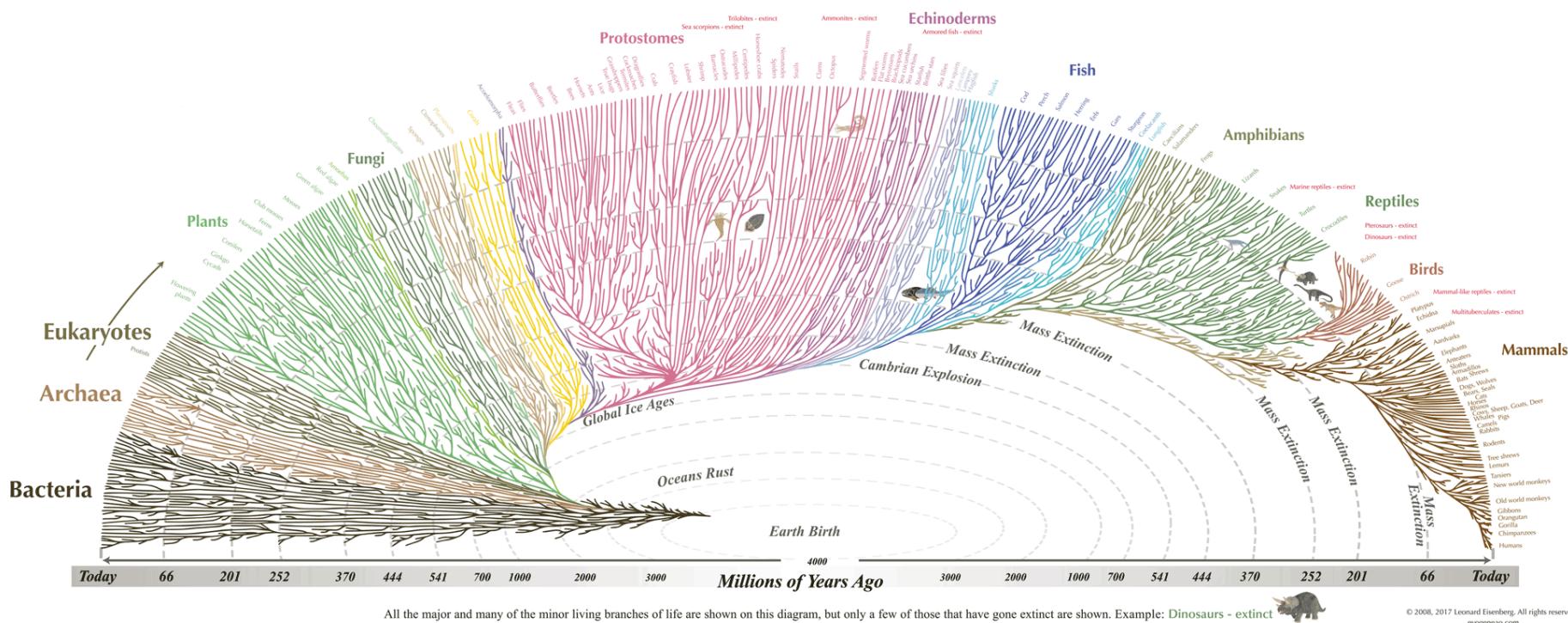
... many other variations

Network layouts: *Trees*



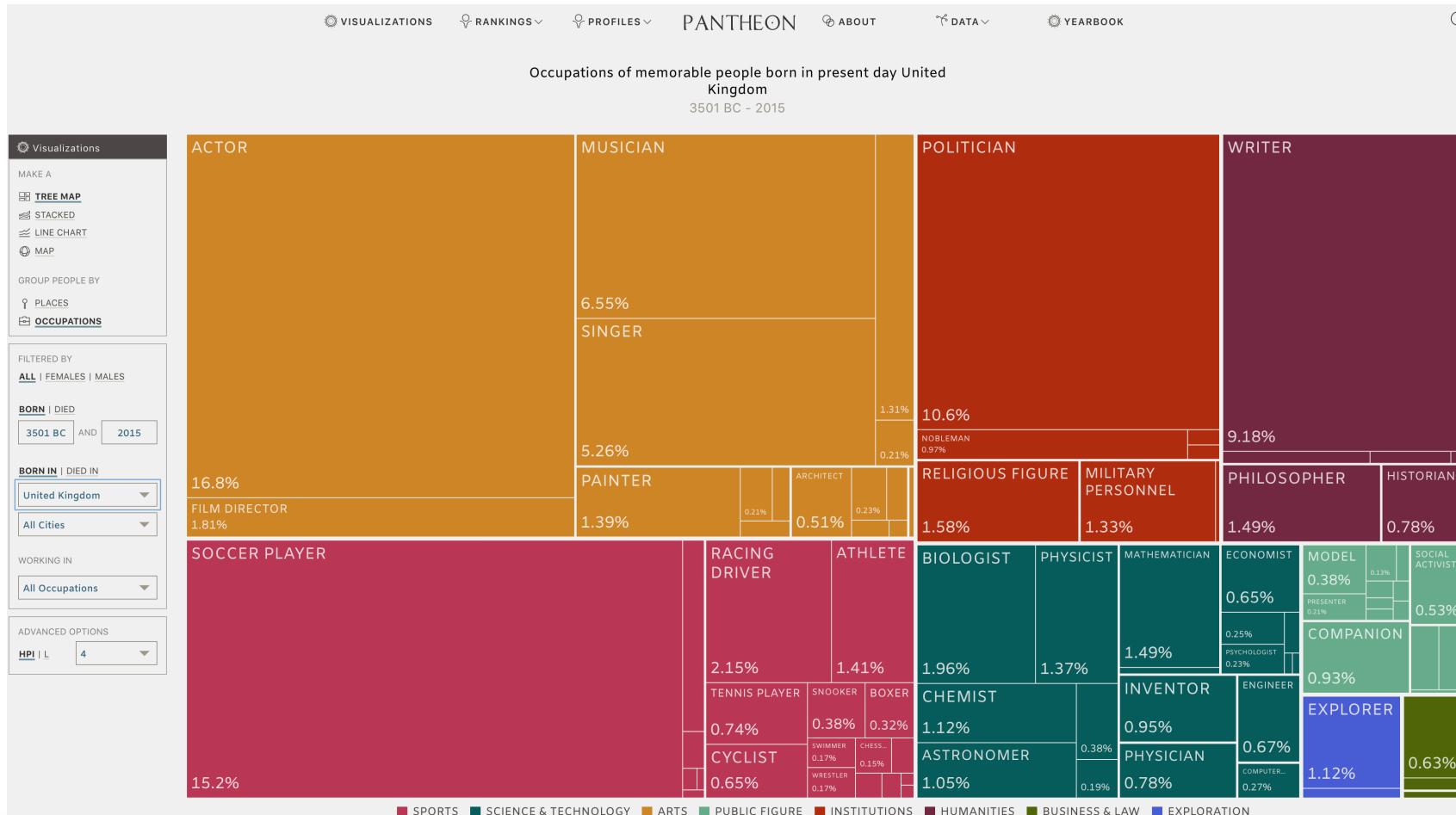
Source: 1296 Ramon Llull's *arbor scientiae* (tree of science)

Network layouts: *Trees*



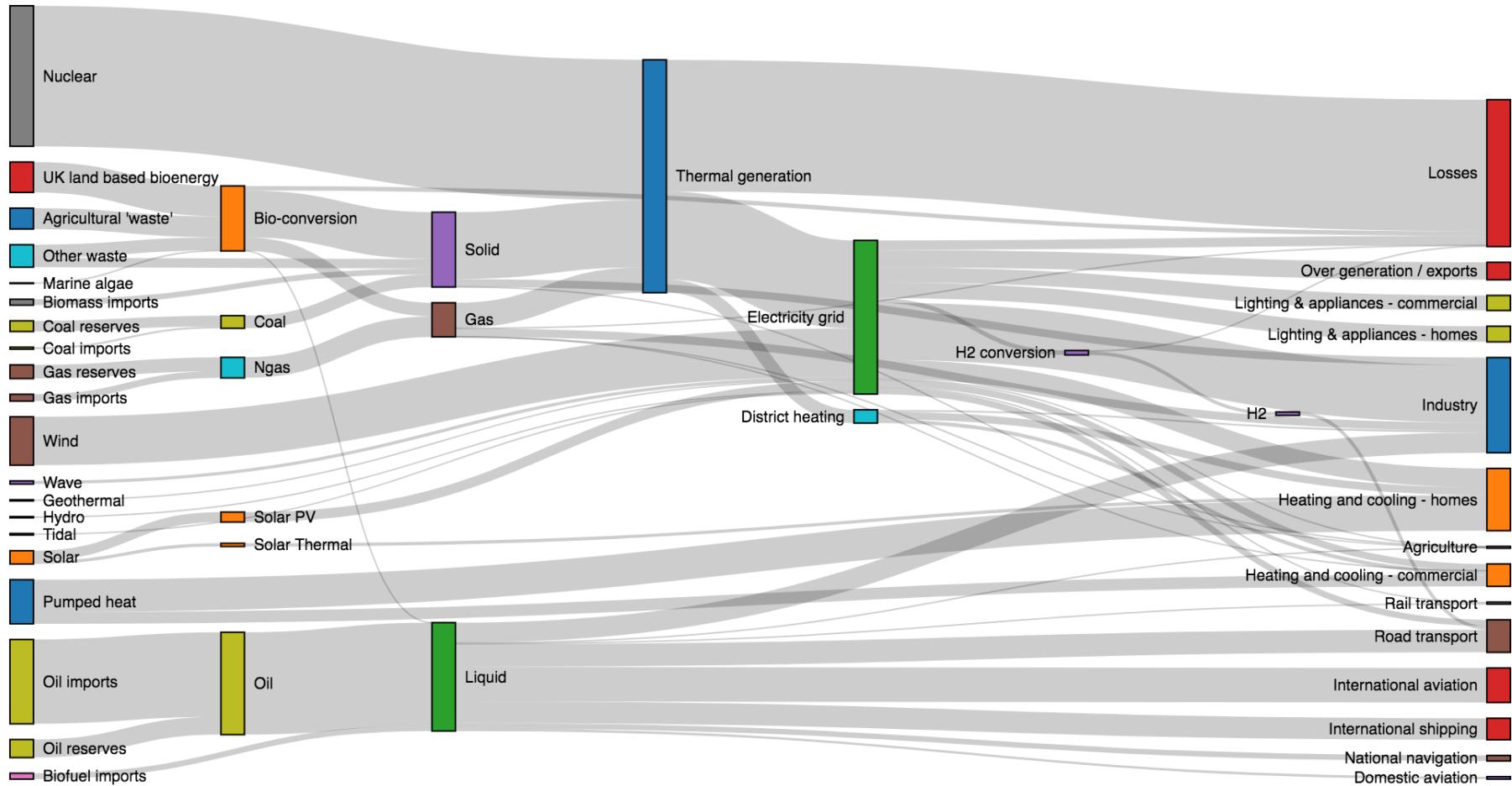
Source: Tree of life (www.evogeneao.com)

Network layouts: *Tree maps*



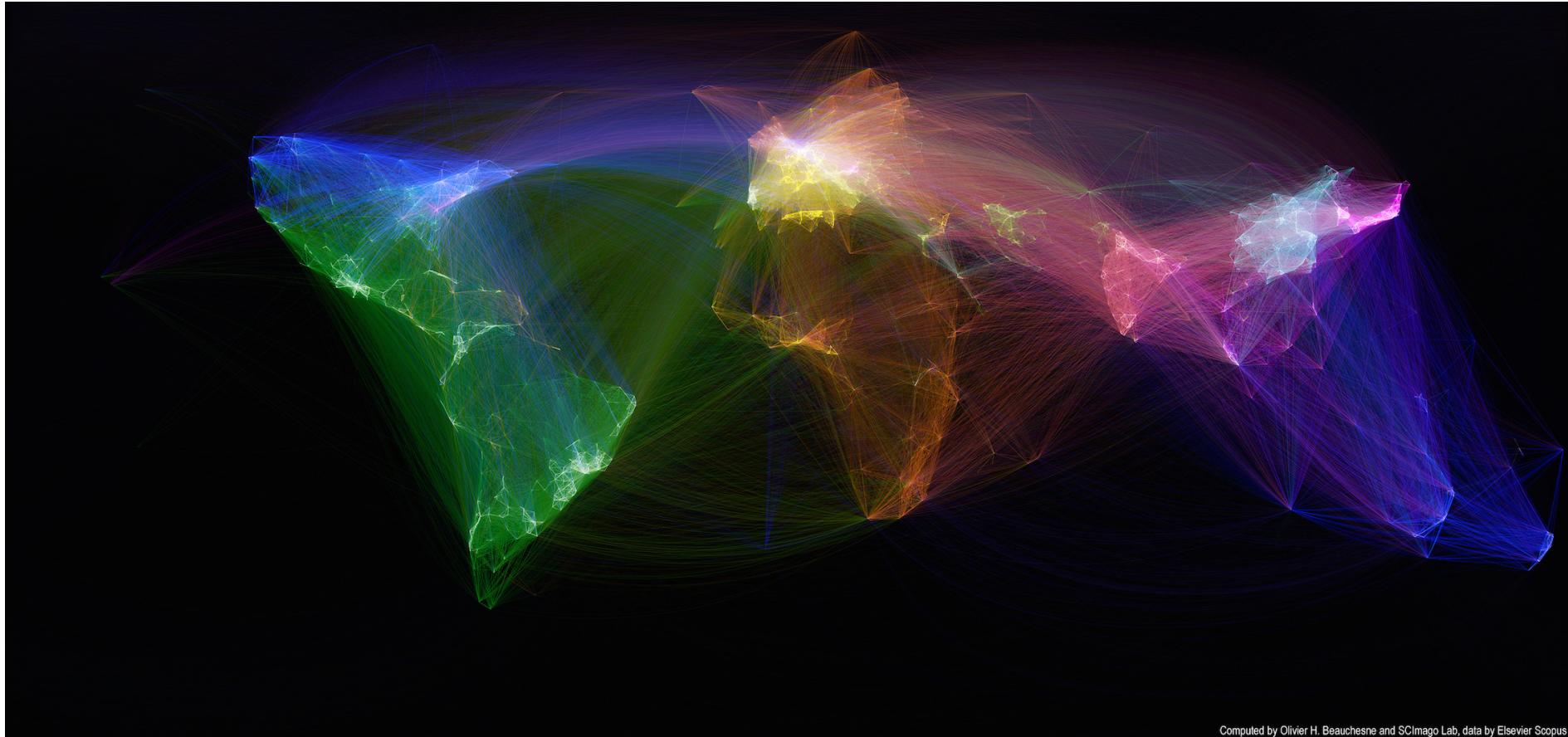
Source: <https://pantheon.world>

Network layouts: Sankey diagrams



Source: UK energy production and consumption in 2050 ([Department of Energy & Climate Change](#))

Network layouts: *Geographical networks*



Computed by Olivier H. Beauchesne and SCImago Lab, data by Elsevier Scopus

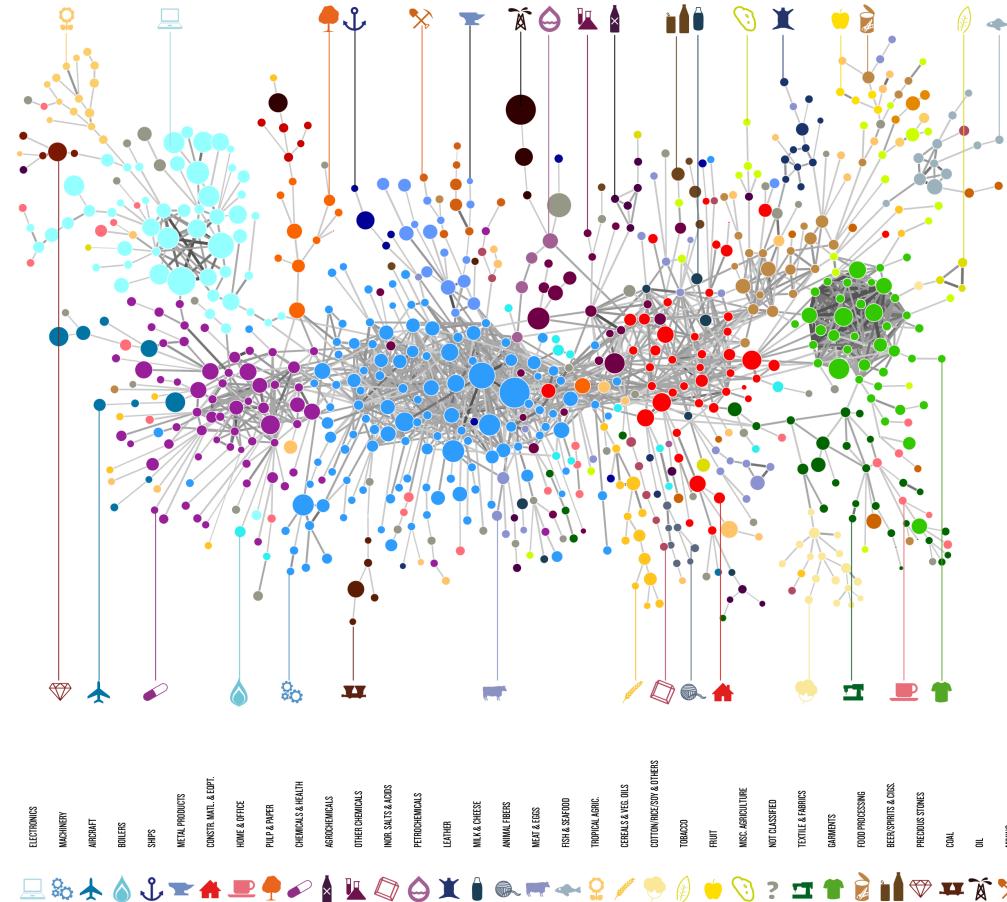
Source: Co-authorship at the city level (SCOPUS 2008-2012) (<http://olihb.com>)

Network layouts: *Citation networks*



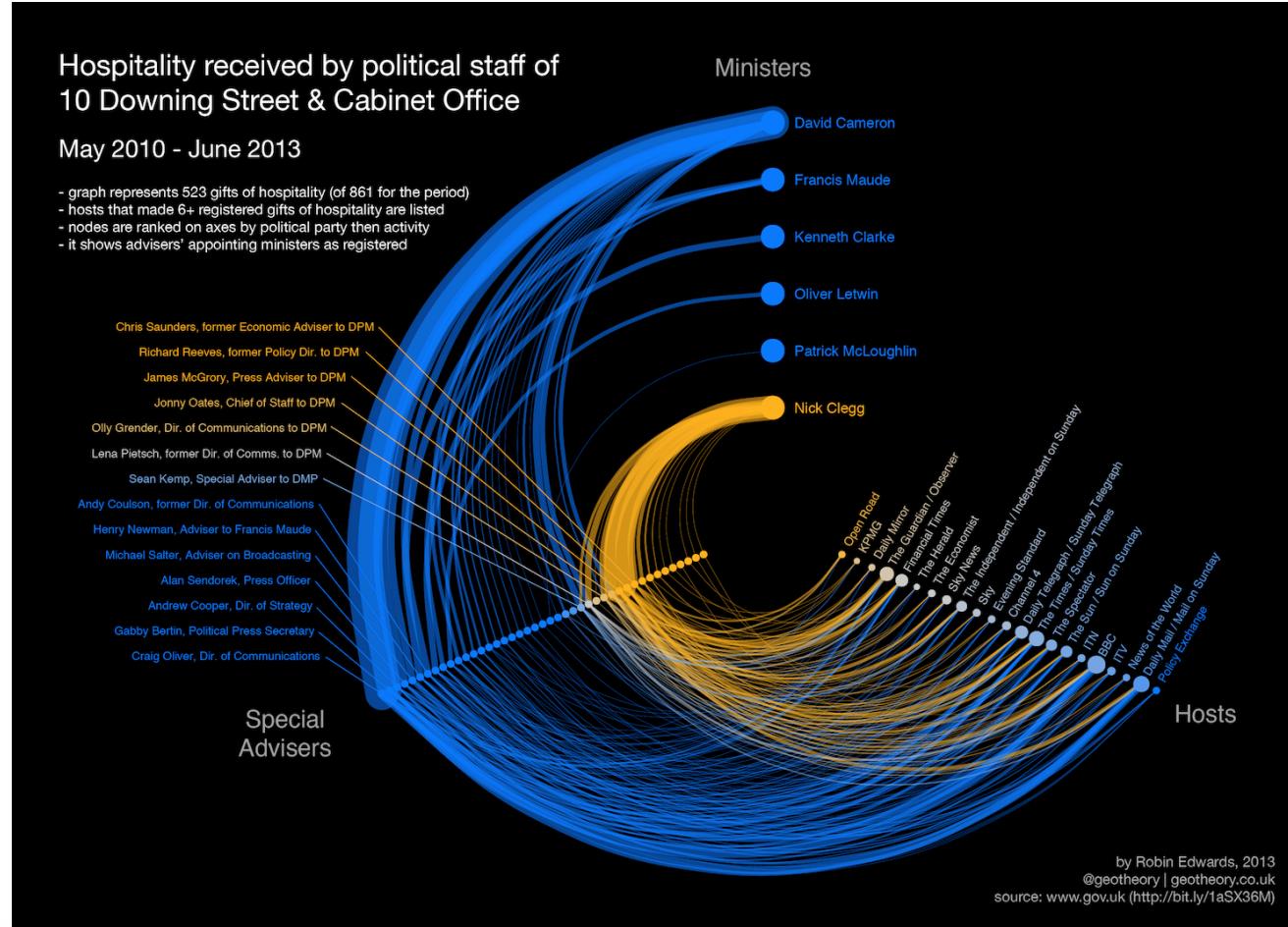
Source: [Nature, November 2019](#)

Network layouts: Networks



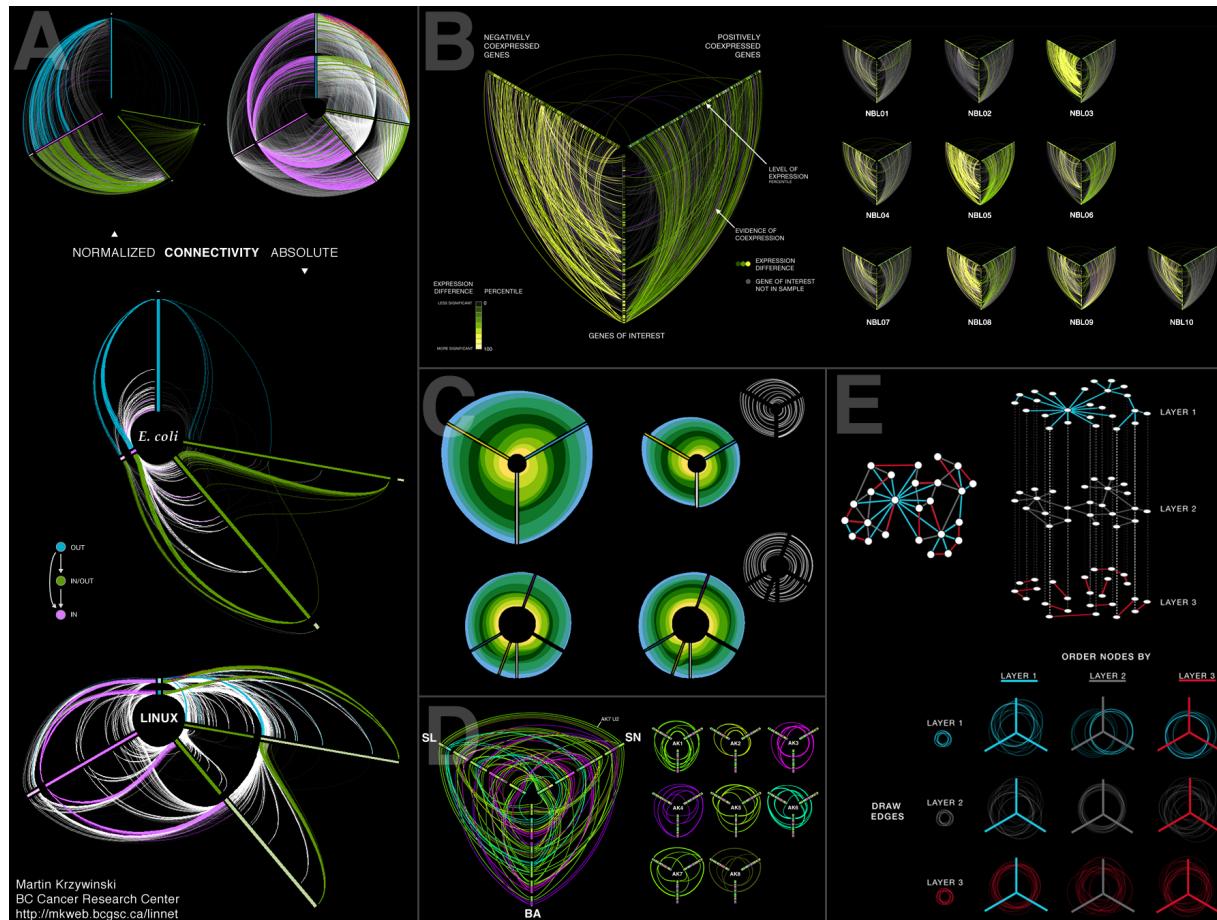
Source: Product space (Hausmann et al. 2013)

Network layouts: *Hive plots*



Source: <https://www.gov.uk/government/collections/special-advisers-transparency-publications>

Network layouts: *Hive plots*



Source: <http://hiveplot.com>

Network layouts: ...

... many other variations

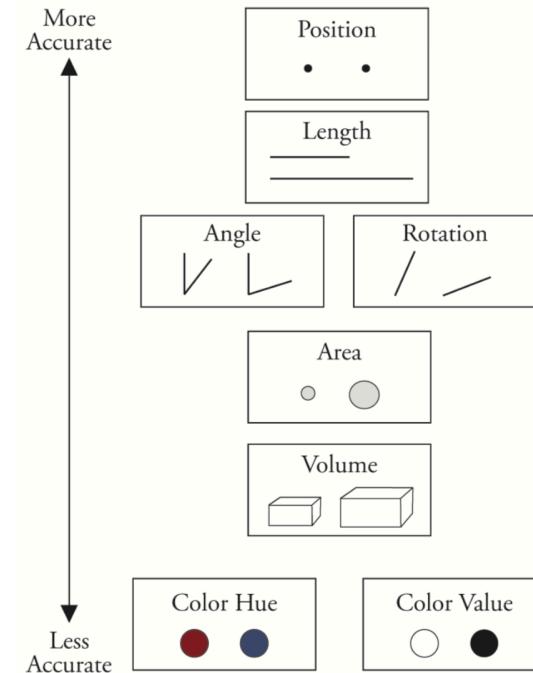
Graphical symbols and variables

Graphical symbols

- Graphical symbols are graphical representations of data information
- Types
 - *Geometric*: point, line, area, surface, volume
 - *Linguistic*: text, numerals, punctuation marks
 - *Pictorial*: images, icons, statistical glyph

Graphical variables

- **Graphic variables** are used to encode additional data
- Types
 - **Spatial:** Position (x, y , z)
 - **Retinal:**
 - *Form:* size, shape, rotation, curvature, angle, closure,
 - *Color:* value, hue, saturation ([colorbrewer](#))
 - *Texture:* spacing, pattern, granularity, gradient
 - *Optics:* blur, transparency, shading
 - *Motion:* speed, velocity, rhythm



Source: Perception Accuracy (Börner 2016)

Graphical symbols and variables

Graphic Variable Types Versus Graphic Symbol Types

		Geometric Symbols						Linguistic Symbols Text, Numerals, Punctuation Marks		Pictorial Symbols Images, Icons, Statistical Glyphs	
		Point	Line	Area	Surface	Volume		Text	Text		
Spatial	x quantitative										
	y quantitative										
	z quantitative										
	Size quantitative	NA (Not Applicable)									
	Shape qualitative	NA									
	Rotation quantitative	NA									
	Curvature quantitative	NA									
	Angle quantitative	NA									
	Closure quantitative	NA						Some table cells are left blank to encourage future exploration of combinations.			
Retinal	Value quantitative										
	Hue qualitative										
	Saturation quantitative										

Source: [Atlas of Science](#) (Börner 2016)

Graphical symbols and variables

Graphic Variable Types Versus Graphic Symbol Types (continued)

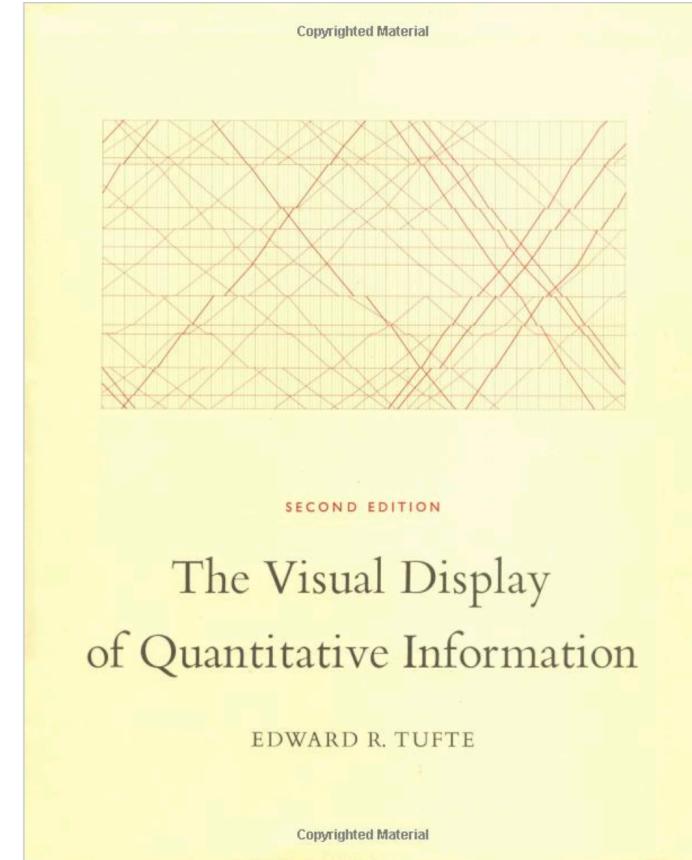
		Geometric Symbols										Linguistic Symbols					Pictorial Symbols					
		Point		Line		Area			Surface			Volume			Text, Numerals, Punctuation Marks			Images, Icons, Statistical Glyphs				
Texture	Spacing	quantitative																				
	Granularity	quantitative																				
	Pattern	qualitative																				
	Orientation	quantitative	NA																			
	Gradient	quantitative																				
	Blur	quantitative																				
	Transparency	quantitative																				
	Shading	quantitative																				
	Stereoscopic Depth	quantitative	Point in foreground ... background	Line in foreground ... background	Area in foreground ... background	Surface in foreground ... background	Volume in foreground ... background	Text in foreground ... background	Numerals in foreground ... background	Punctuation marks in foreground ... background	Icons in foreground ... background											
	Speed	quantitative																				
Motion	Velocity	quantitative																				
	Rhythm	quantitative	Blinking point slow ... fast	Blinking line slow ... fast	Blinking area slow ... fast	Blinking surface slow ... fast	Blinking volume slow ... fast	Blinking text slow ... fast	Blinking icons slow ... fast													

Source: [Atlas of Science](#) (Börner 2016)

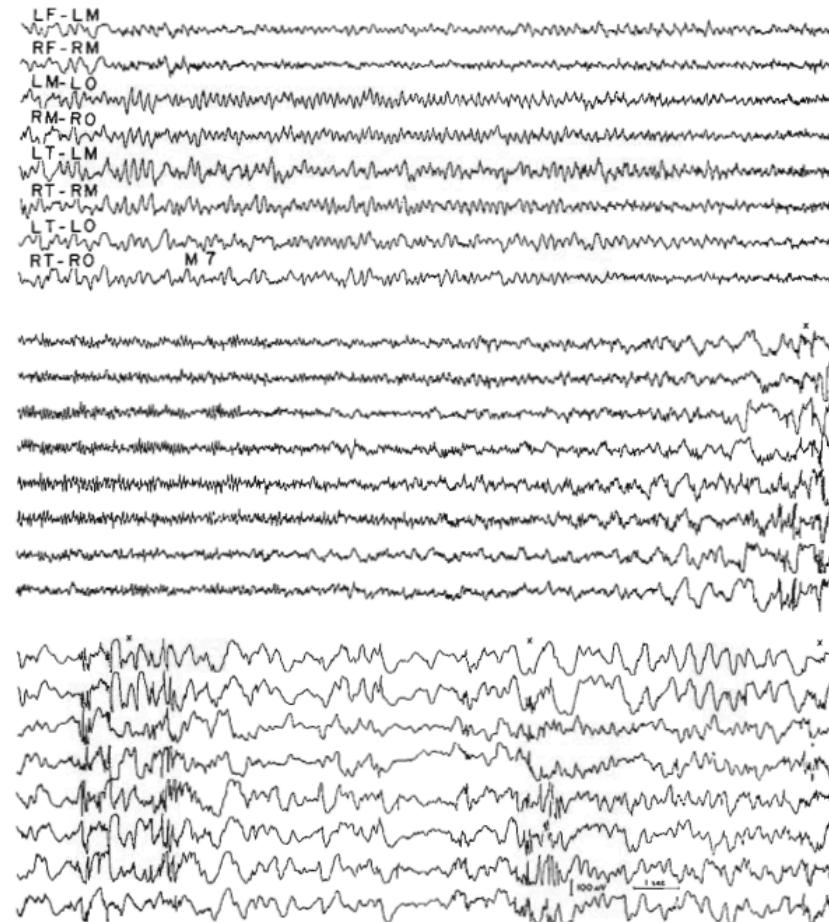
Data-ink ratio

The concept of **data-ink ratio** has been introduced by Tufte (2001)

- Most of the 'ink' used to print a graphic should represent data-information
- **Data-ink** is defined as the part of the graphic that we cannot erase without losing information
- **Data-ink ratio** is defined as the proportion of a graphic's ink devoted to the display of non-redundant information (data-ink)

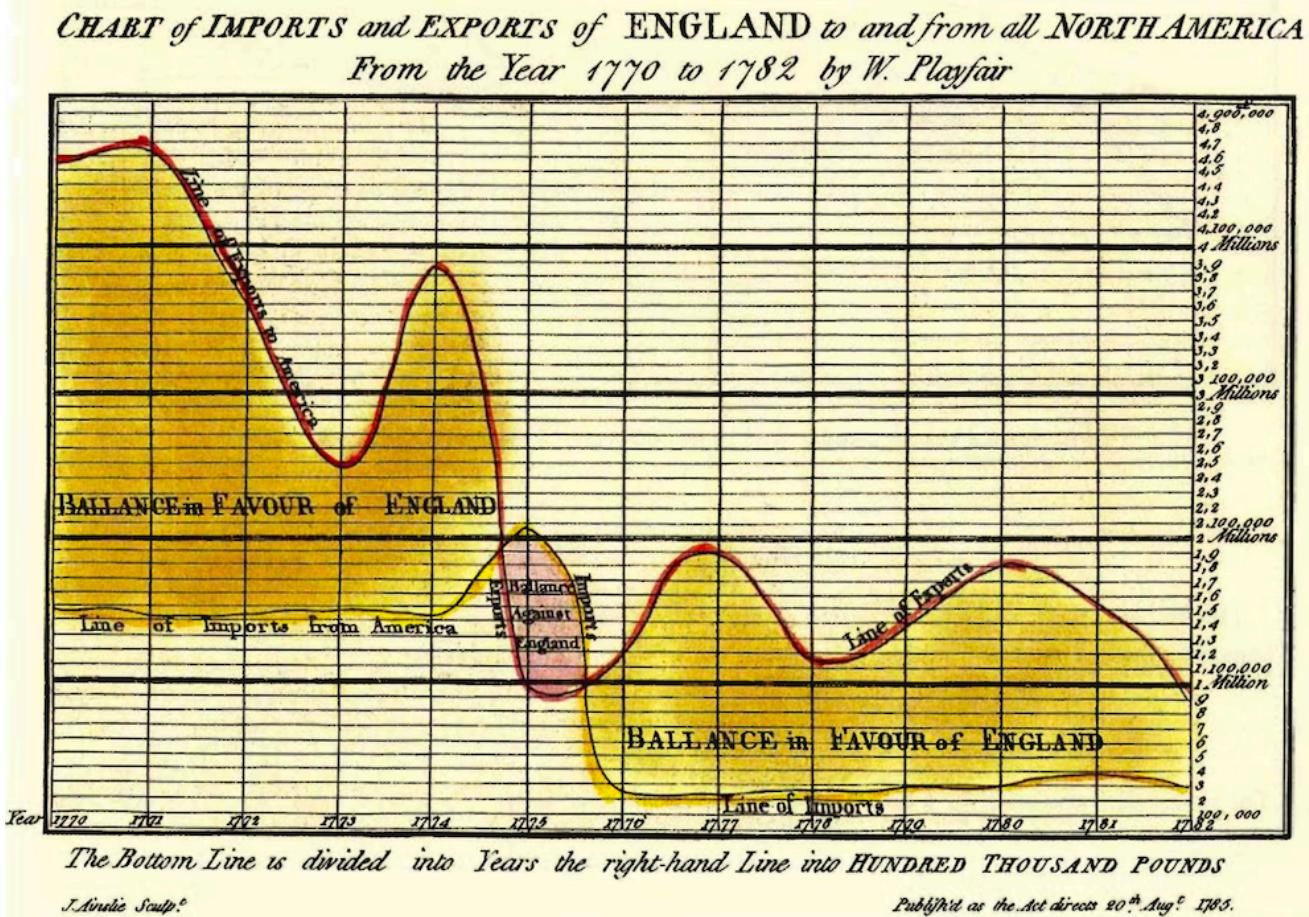


Data-ink ratio



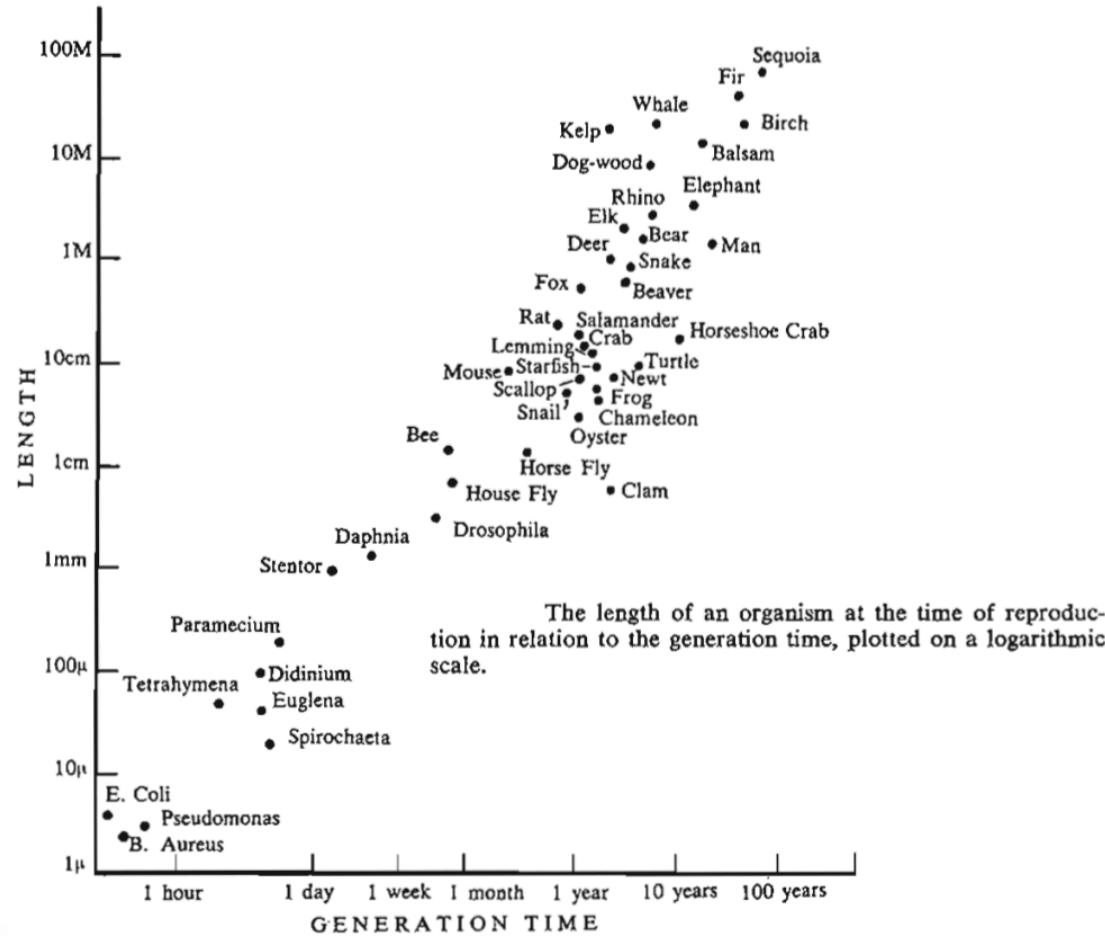
Source: Fundamentals of electroencephalography (Tufte 2001)

Data-ink ratio



Source: England's import/export from and to North America from 1770 to 1782 (Tufte 2001)

Data-ink ratio



Source: John Tyler Bonner. 1975. Size and Cycle: An Essay on the Structure of Biology, Princeton Legacy Library (Tufte 2001)

Beyond statistics and visualisation

Triggering emotions

- Raw data do not tell as much
- Statistics and visualisation help us to go beyond raw data to see patterns, and relationships between variables
- Data visualisation helps us to communicate this into a story and to trigger emotions and actions
- It is crucial to consider your audience

Triggering emotions

- Watch Hans Rosling's videos
- BBC: [200 years in 4 minutes](#)
- TED talk: [Global Population Growth, Box by Box](#)
- TED talk: [Asia's Rise = How and When](#)



Source: [New York Times](#), 9 February 2017

Graphical integrity

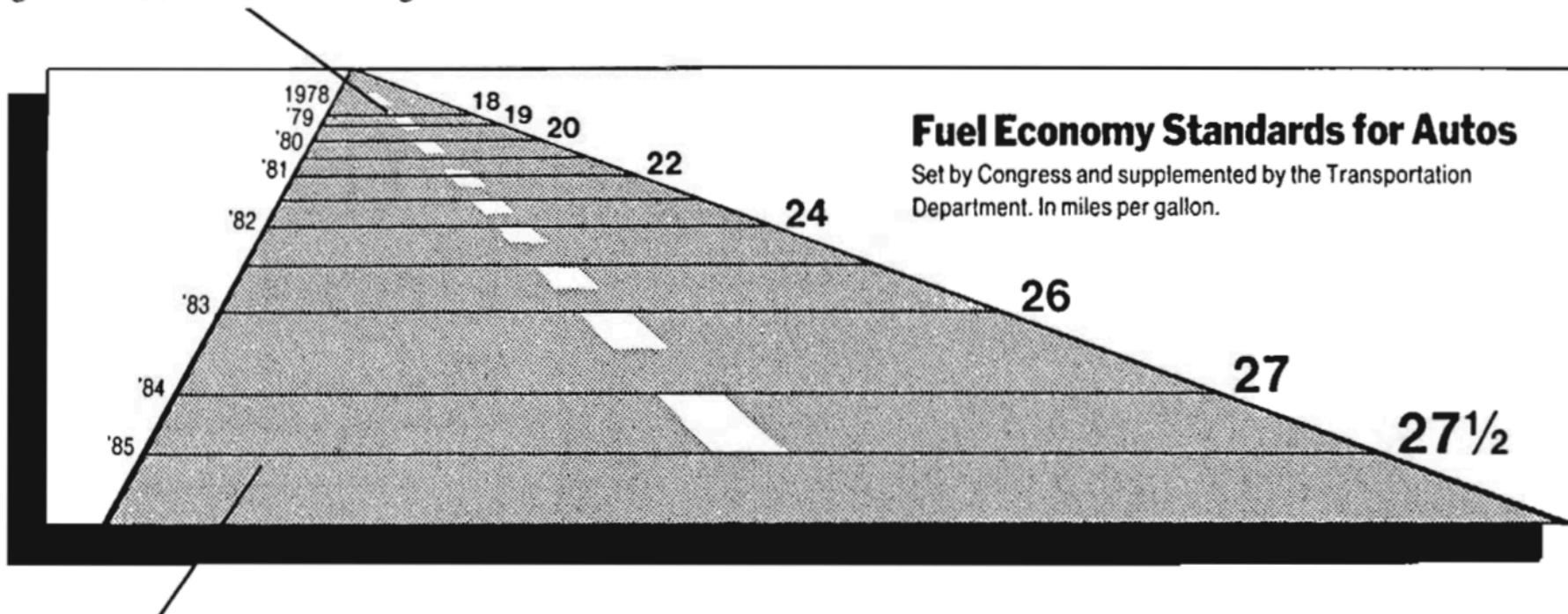
- Graphics are a form of **data communication**
- Graphics may (**intentionally or unintentionally**) distort this communication and therefore the underlying data
- The **visual perception** has to be assessed carefully

$$\text{Lie factor} = \frac{\text{Size of effect shown in graphic}}{\text{Size of effect in data}}$$

- A **lie factor** greater than 1.05 or smaller than 0.95 introduces a significant distortion

Graphical integrity

This line, representing 18 miles per gallon in 1978, is 0.6 inches long.



This line, representing 27.5 miles per gallon in 1985, is 5.3 inches long.

Source: New York Times, August 9, 1978 (Tufte 2001)

Graphical integrity

Size of effect in data = $(27.5 - 18.0)/18.0 = 0.53$

Size of effect shown in graphic = $(5.3 - 0.6)/0.6 = 0.783$

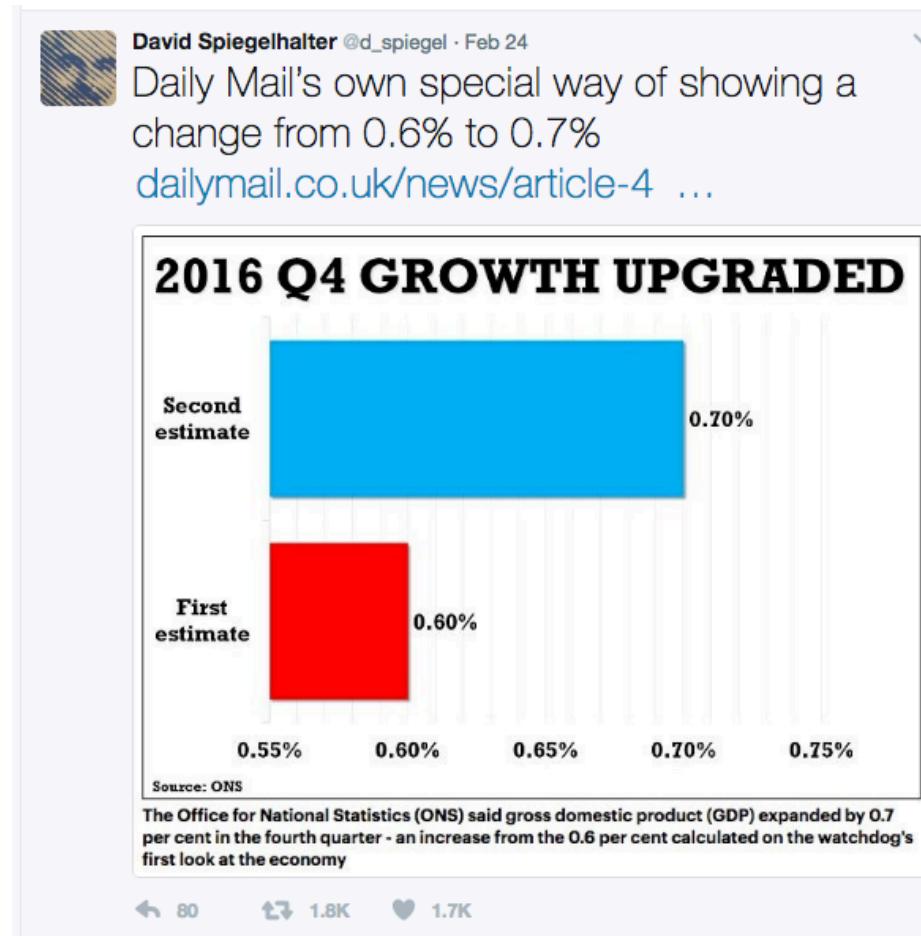
Lie factor = $0.783/0.53 = 14.8$

Graphical integrity



Source: Campaign ad in Islington South (<https://electionleaflets.org/brb.html>)

Graphical integrity



Source: Twitter (March, 2017)

Data visualisation in R

Data visualisation in R

- The most popular package is `ggplot2`
- Developed by Hadley Wickham and colleagues
- It is based on the **Grammar of Graphics**: components that can be composed in different manners (like a LEGO)
- It works in **layers of graphics**
- Layers map data into aesthetic attributes (e.g. color, shape, size) of geometric objects (e.g. points, lines, bars), or in other words **graphical symbols and variables**



Elements of `ggplot2`

- Data to visualise and a set of aesthetic (`aes`) mappings (e.g. `x`, `y`, `group`)
- Layers of geometric elements (`geom`) and statistical transformation (`stat`)
- Scales which allow us to read the data from the plot (legend and axis)
- A system of coordinates (e.g. coordinates of observations, gridlines)
- Faceting to describe how data are separated in different subsets
- Theme (e.g. size, background colors)

Elements of ggplot2

- | | |
|------------------------------------|-------------|
| Describes all the non-data ink | Theme |
| Plotting space for the data | Coordinates |
| Statistical models & summaries | Statistics |
| Rows and columns of sub-plots | Facets |
| Shapes used to represent the data | Geometries |
| Scales onto which data is mapped | Aesthetics |
| The actual variables to be plotted | Data |



Source: <https://api.rpubs.com/NemiVoraR/511391>

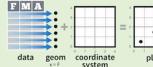
ggplot2 cheat sheet

Data Visualization with ggplot2

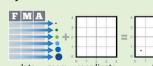


Basics

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



To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** and **y** locations.



Build a graph with **qplot()** or **ggplot()**

aesthetic mappings **data** **geom**

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

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

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

Begins a plot that you finish by adding layers to. No defaults, but provides more control than qplot().

```
data +  
  geom_point(aes(color = cyl)) +  
  geom_smooth(method = "lm") +  
  coord_cartesian() +  
  scale_color_gradient() +  
  theme_bw()
```

Add a new layer to a plot with a **geom_***() or **stat_***() function. Each provides a geom, a set of aesthetic mappings, and a default stat and position adjustment.

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.

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com

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

One Variable

Continuous

```
a + geom_area(stat = "bin")  
x, y, alpha, color, fill, linetype, size  
b + geom_area(aes(y = ..density..), stat = "bin")  
a + geom_density(kernel = "gaussian")  
x, y, alpha, color, fill, linetype, size, weight  
b + geom_density(aes(y = ..count..))
```

Dot Plot

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

Frequency Polygon

```
a + geom_freqpoly()  
x, y, alpha, color, linetype, size  
b + geom_freqpoly(aes(y = ..density..))
```

Histogram

```
a + geom_histogram(binwidth = 5)  
x, y, alpha, color, fill, linetype, size, weight  
b + geom_histogram(aes(y = ..density..))
```

Discrete

```
b <- ggplot(mpg, aes(f1))
```

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

Graphical Primitives

Continuous Y

```
c <- ggplot(map, aes(long, lat))  
c + geom_polygon(aes(group = group))  
x, y, alpha, color, fill, linetype, size
```

```
d <- ggplot(economics, aes(date, unemployed))
```

```
d + geom_path(linewidth = "butt",  
             linjoin = "round", linmitre = 1)  
x, y, alpha, color, linetype, size
```

```
d + geom_ribbon(aes(ymin = unemployed - 900,  
                     ymax = unemployed + 900))  
x, y, alpha, color, fill, linetype, size
```

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

```
e + geom_segment(aes(  
                  xend = long + delta_long,  
                  yend = lat + delta_lat))  
x, xend, y, yend, alpha, color, linetype, size
```

```
e + geom_rect(aes(xmin = long, ymin = lat,  
                   xmax = long + delta_long,  
                   ymax = lat + delta_lat))  
xmax, xmin, ymax, ymin, alpha, color, fill,
```

linetype, size

Discrete X, Continuous Y

```
f <- ggplot(diamonds, aes(cut, color))
```

```
f + geom_jitter()  
lower, middle, upper, x, y, ymin, alpha,
```

color, fill, linetype, shape, size, weight

```
g + geom_dotplot(binaxis = "y",  
                 stackdir = "center")  
x, y, alpha, color, fill
```

```
g + geom_violin(scale = "area")  
x, y, alpha, color, fill, linetype, size, weight
```

Discrete X, Discrete Y

```
h <- ggplot(diagrams, aes(cut, color))
```

```
h + geom_jitter()  
x, y, alpha, color, fill, shape, size
```

Two Variables

Continuous X, Continuous Y

```
f <- ggplot(mpg, aes(cty, hwy))
```

```
f + geom_blank()  
x, y, alpha, color, fill, linetype, size, weight
```

```
f + geom_hex()  
x, y, alpha, colour, fill, size
```

```
i + geom_hex2d(binwidth = c(5, 0.5))  
xmax, xmin, ymax, ymin, alpha, color, fill,
```

linetype, size, weight

Continuous Bivariate Distribution

```
j <- ggplot(movies, aes(year, rating))
```

```
i + geom_hex2d(binwidth = c(5, 0.5))  
x, y, alpha, colour, fill, size
```

Continuous Function

```
j <- ggplot(economics, aes(date, unemployed))
```

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

```
j + geom_line()  
x, y, alpha, color, linetype, size
```

```
j + geom_step(direction = "hv")  
x, y, alpha, color, linetype, size
```

Visualizing error

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

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

```
k + geom_crossbar(fatten = 2)  
x, y, ymax, ymin, alpha, color, fill, linetype, size
```

```
k + geom_errorbar()  
x, ymax, ymin, alpha, color, linetype, size, width (also geom_errorbarh())
```

```
k + geom_linerange()  
x, ymin, ymax, alpha, color, linetype, size
```

```
k + geom_pointrange()  
x, y, ymin, ymax, alpha, color, fill, linetype, size
```

Maps

```
data <- data.frame(murder = USArrests$Murder,
```

```
state = tolower(rownames(USArrests)))
```

```
map <- map_data("state")
```

```
[<- ggplot(data, aes(state = murder))
```

```
[+ geom_map(aes(state_id = state), map = map) +
```

```
expand_limits(x = map$long, y = map$lat)
```

map_id, alpha, color, fill, linetype, size

Three Variables

```
m + geom_raster(aes(fill = z), hjust = 0.5,
```

vjust = 0.5, interpolate = FALSE)

x, y, alpha, fill

```
m + geom_contour(aes(z = z))  
x, y, z, alpha, colour, linetype, size, weight
```

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

Source: <https://rstudio.com/resources/cheatsheets>

Learn more at docs.ggplot2.org • ggplot2 0.9.3.1 • Updated: 3/15

ggplot2 cheat sheet

Stats - An alternative way to build a layer

Some plots visualize a **transformation** of the original data set. Use a **stat** to choose a common transformation to visualize, e.g. `a + geom_bar(stat = "bin")`

```

graph LR
    data --> stat[stat]
    stat --> geom[geom]
    geom --> coord[coordinate system]
    coord --> plot[plot]
  
```

Each stat creates additional variables to map aesthetics to. These variables use a common `..name..` syntax.

stat functions and geom functions both combine a stat with a geom to make a layer, i.e. `stat_bin(geom="bar")` does the same as `geom_bar(stat="bin")`

stat function	layer specific mappings	variable created by transformation
<code>i + stat_density2d(aes(fill = ..level..), geom = "polygon", n = 100)</code>	<code>geom for layer</code>	<code>parameters for stat</code>

```

a + stat_bin(binwidth = 1, origin = 10)           1D distributions
x,y | ..count.., ..ncount.., ..density..
a + stat_bindot(binwidth = 1, binaxis = "x")
x,y | ..count..
a + stat_density(adjust = 1, kernel = "gaussian")
x,y | ..density.., ..scaled..
f + stat_binned(pins = 30, drop = TRUE)          2D distributions
x,y | ..density..
f + stat_binned(bins = 30)
x,y, fill | ..count.., ..density..
f + stat_density2d(contour = TRUE, n = 100)
x,y, color, size | ..level..
m + stat_contour(besize = 2)                      3 Variables
x,y,z | ..density..
m + stat_spoke(aes(radius = angle = z))
angle, radius, x, end_x, y_end | ..xend.., ..yend..
m + stat_summary_hex(besize = 2, bins = 30, fun = mean)
x,y,z, fill | ..value..
m + stat_summary2d(aes(z = z), bins = 30, fun = mean)
x,y,z, fill | ..value..
g + stat_boxplot(coef = 1.5)                     Comparisons
x,y | ..lower.., ..middle.., ..upper.., ..outliers..
g + stat_density2d(adjust = 1, kernel = "gaussian", scale = "area")
x,y | ..density.., ..scaled.., ..count.., ..n.., ..width..
f + stat_ecdf(n = 40)                            Functions
x,y | ..x.., ..y..
f + stat_ecdf(aes(quartiles = c(0.25, 0.5, 0.75), formula = y ~ log(x), method = "nd"))
x,y | ..quartile.., ..x.., ..y..
f + stat_smooth(method = "auto", formula = y ~ x, se = TRUE, n = 80, fullrange = FALSE, level = 0.95)
x,y | ..se.., ..x.., ..y.., ..ymin.., ..ymax..
ggplot() + stat_function(aes(x = 3.3))           General Purpose
fun = dnorm, n = 101, args = list(sd = 0.5)
x,y | ..x.., ..y..
f + stat_identity()
ggplot() + stat_qq(aes(sample = 1:100), distribution = qt, dparams = list(df = 5))
sample, x,y | ..x.., ..y..
f + stat_sum()
x,y | ..sum..
f + stat_summary(fun.data = "mean_cl_boot")
f + stat_unique()
  
```

Scales

Scales control how a plot maps data values to the visual values of an aesthetic. To change the mapping, add a custom scale.

n + b + geom_bar(aes(fill = f))	scale	aesthetic	prepackaged	scale to use	scale specific arguments
<code>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"))</code>	range of values to include in mapping	title to use in legend/axis	labels to use in legend/axis	breaks to use in legend/axis	

General Purpose scales

Use with any aesthetic:

- `alpha`, `color`, `fill`, `linetype`, `shape`, `size`
- `scale_*_continuous()` - map cont. values to visual values
- `scale_*_discrete()` - map discrete values to visual values
- `scale_*_identity()` - use data values as visual values
- `scale_*_manual(values = c(l))` - map discrete values to manually chosen visual values

X and Y location scales

Use with x or y aesthetics (x shown here)

- `scale_x_date(labels = date_format("%m/%d"), breaks = date_breaks("2 weeks"))` - treat x values as dates. See `strptime` for label formats.
- `scale_x_datetime()` - treat x values as date times. Use same arguments as `scale_x_date()`.
- `scale_x_log10()` - Plot x on log10 scale
- `scale_x_reverse()` - Reverse direction of x axis
- `scale_x_sqrt()` - Plot x on square root scale

Color and fill scales

Discrete	Continuous
<code>n <- b + geom_bar(aes(fill = f))</code>	<code>o <- a + geom_dotplot(aesthetics = x..)</code>
<code>n + scale_fill_brewer(palette = "Blues")</code>	<code>o + scale_fill_gradient(low = "white", high = "yellow")</code>
For palette choices, see RColorBrewer display.brewer.all()	<code>o + scale_fill_gradient2(midpoint = "white", mid = "white", midpoint = 25)</code>
<code>n + scale_fill_grey(start = 0.2, end = 0.8, no.value = "rot")</code>	<code>o + geom_bar(position = "stack")</code>
	<code>Stack elements on top of one another, normalize height</code>
	<code>s + geom_bar(position = "fill")</code>
	<code>Stack elements on top of one another</code>
	<code>s + geom_bar(position = "stack")</code>
	<code>Stack elements on top of one another</code>
	<code>f + geom_point(position = "jitter")</code>
	<code>Add random noise to X and Y position of each element to avoid overplotting</code>

Shape scales

Manual shape values	ggplot2
<code>p <- f + geom_point(aes(shape = m))</code>	<code>0 ◊ 6 ▽ 12 □ 18 ◆ 24 ▲</code>
<code>p + scale_shape(solid = FALSE)</code>	<code>1 ○ 7 △ 13 ▨ 19 ● 25 ▼</code>
<code>p + scale_shape_manual(values = m)</code>	<code>2 △ 8 ▴ 14 □ 20 ● 26 ▲</code>
Shape values shown in chart on right	<code>3 △ 9 ▽ 15 ■ 21 ▲ 27 ▲</code>
	<code>4 × 10 △ 16 ● 22 □ 28 ▲</code>
	<code>5 ◊ 11 □ 17 ▲ 23 ▲</code>
	<code>6 ◊ 13 △ 19 ▨ 25 ▲</code>

Size scales

ggplot2	geom_size_area(max = 6)
<code>q <- f + geom_point(aes(size = cyl))</code>	<code>Value mapped to area of circle (not radius)</code>

Coordinate Systems

r < b + geom_bar()	coord_cartesian(xlim = c(0, 5))
	xlim, ylim
	The default cartesian coordinate system
<code>r + coord_fixed(ratio = 1/2)</code>	ratio, xlim, ylim
	Cartesian coordinates with fixed aspect ratio between x and y units
<code>r + coord_flip()</code>	ylim, xlim
	Flipped Cartesian coordinates
<code>r + coord_polar(theta = "x", direction = 1)</code>	theta, start, direction
	Polar coordinates
<code>r + coord_trans(xtrans = "sqrt")</code>	xtrans, ytrans, xlim, ylim
	Transformed cartesian coordinates. Set extras and strains to the name of a window function.
<code>z + coord_map(projection = "ortho", orientation = c(41, -74, 0))</code>	projection, orientation, xlim, ylim
	Map projections from the mapproj package (mercator (default), azequalarea, lagrange, etc.)

Faceting

Facets divide a plot into subplots based on the values of one or more discrete variables.

t + facet_grid(~ fl)	t + facet_grid(year ~ .)	t + facet_grid(year ~ fl)	t + facet_wrap(~ fl)
facet into columns based on fl	facet into rows based on year	facet into both rows and columns	wrap facets into a rectangular layout

Set scales to let axis limits vary across facets

t + facet_grid(~ x, scales = "free")
x and y axis limits adjust to individual facets
* <code>"free_x"</code> - x axis limit adjust
* <code>"free_y"</code> - y axis limit adjust

Set labeller to adjust facet labels

t + facet_grid(~ fl, labeller = label_both)	t + facet_grid(~ fl, labeller = label_bquote(alpha^.x..))	t + facet_grid(~ fl, labeller = label_parsed(c, d, e, p, r))
fl: d	fl: e	fl: p
fl: r		

Position Adjustments

Position adjustments determine how to arrange geoms that would otherwise occupy the same space.

s <- ggplot(mpg, aes(flf, fill = drv))	s + geom_bar(position = "dodge")
	Arrange elements side by side
	<code>s + geom_bar(position = "fill")</code>
	Stack elements on top of one another, normalize height
	<code>s + geom_bar(position = "stack")</code>
	Stack elements on top of one another
	<code>f + geom_point(position = "jitter")</code>
	Add random noise to X and Y position of each element to avoid overplotting

Labels

t + ggtitle("New Plot Title")	Use scale functions to update legend labels
Add a main title above the plot	
<code>t + xlab("New X label")</code>	Change the label on the X axis
<code>t + ylab("New Y label")</code>	Change the label on the Y axis
<code>t + labs(title = "New title", x = "New x", y = "New y")</code>	All of the above

Legends

t + theme(legend.position = "bottom")	Place legend at "bottom", "top", "left", or "right"
	<code>t + guides(color = "none")</code>
	Set legend type for each aesthetic: colorbar, legend, or none (no legend)
<code>t + scale_fill_discrete(name = "Title", labels = c("A", "B", "C"))</code>	Set legend title and labels with a scale function.

Themes

r + theme_bw()	r + theme_classic()	r + theme_minimal()
White background with grid lines	White background with no gridlines	Minimal theme
<code>ggthemes</code> - Package with additional ggplot2 themes		

Zooming

Without clipping (preferred)	With clipping (removes unseen data points)
<code>t + coord_cartesian(xlim = c(0, 100), ylim = c(10, 20))</code>	
	<code>t + xlim(0, 100) + ylim(10, 20)</code>
	<code>t + scale_x_continuous(limits = c(0, 100)) + scale_y_continuous(limits = c(0, 100))</code>

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com

Learn more at [docs.ggplot2.org](#) • ggplot2 0.9.3.1 • Updated: 3/15

Source: <https://rstudio.com/resources/cheatsheets>

ggplot2 in practice

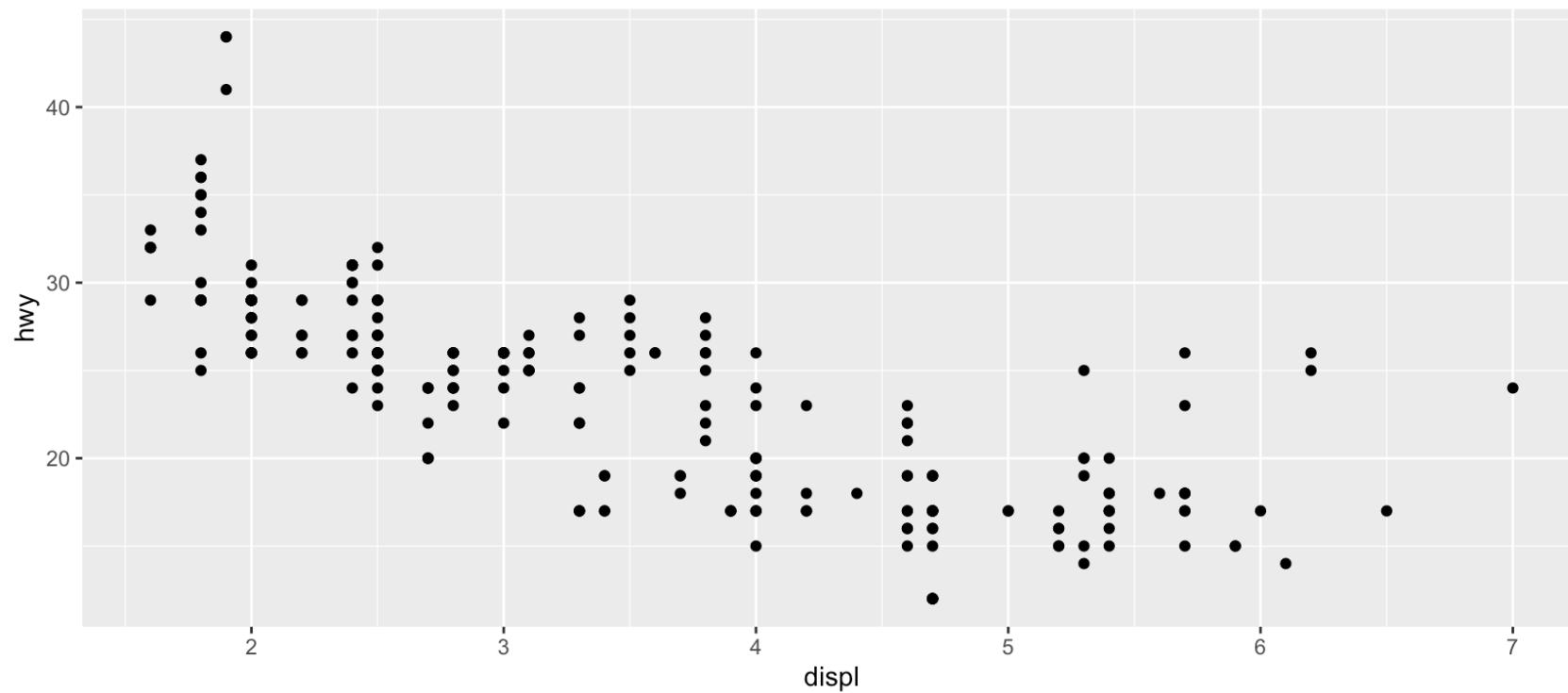
- We focus on the `mpg` dataset: `cty` (mpg for driving), `hwy` (mpg for highway driving), `displ` (engine displacement in liters), `drv` (drivetrain: front wheel [f], rear wheel [r] or four wheel [4]), `model` (car model), `class` (two seater, SUV, compact, etc.)

```
library(ggplot2)
print(mpg, n = 4)
```

```
## # A tibble: 234 x 11
##   manufacturer model displ year cyl trans     drv     cty     hwy fl class
##   <chr>        <chr>  <dbl> <int> <int> <chr>    <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15) f        18     29 p    compa...
## 2 audi         a4      1.8  1999     4 manual(m5) f        21     29 p    compa...
## 3 audi         a4      2.0  2008     4 manual(m6) f        20     31 p    compa...
## 4 audi         a4      2.0  2008     4 auto(av)   f        21     30 p    compa...
## # ... with 230 more rows
```

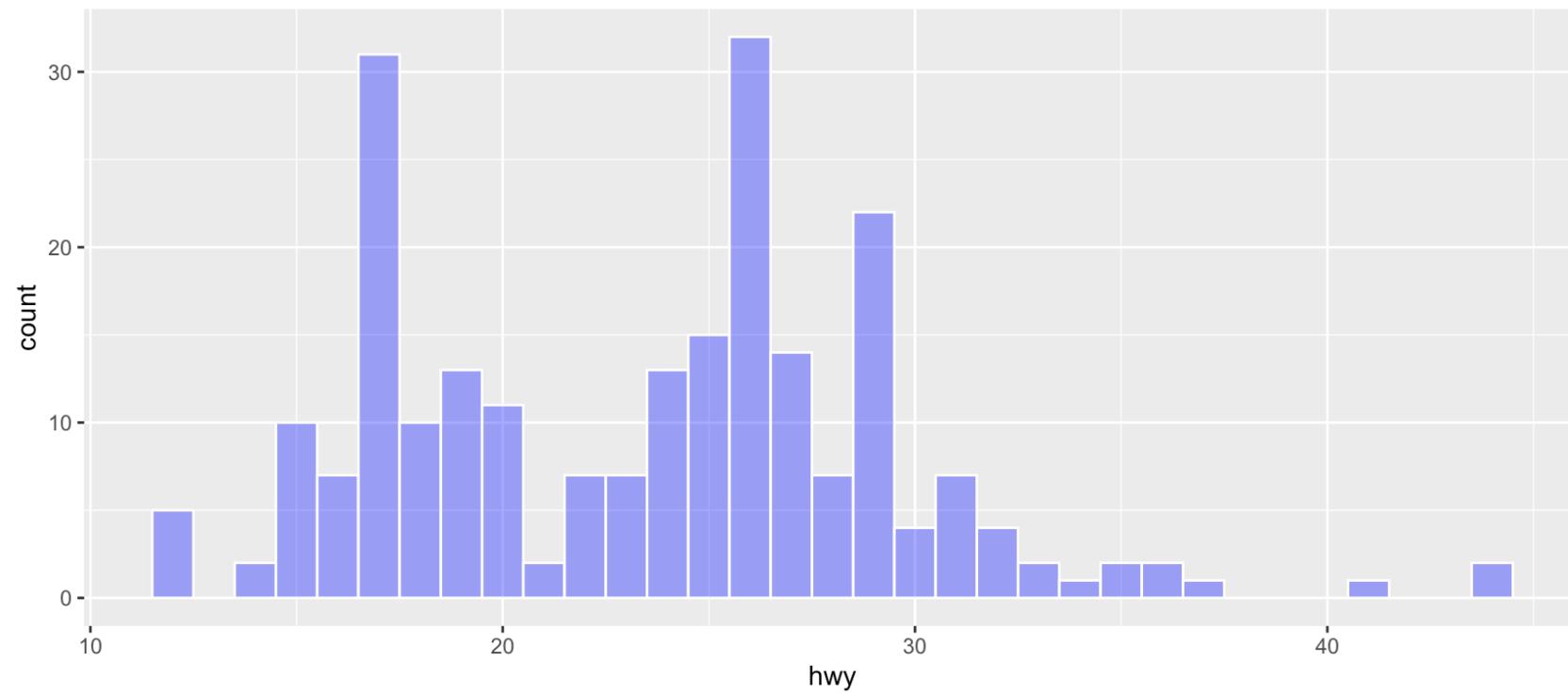
ggplot2 in practice: one variable

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +  
  geom_point()
```



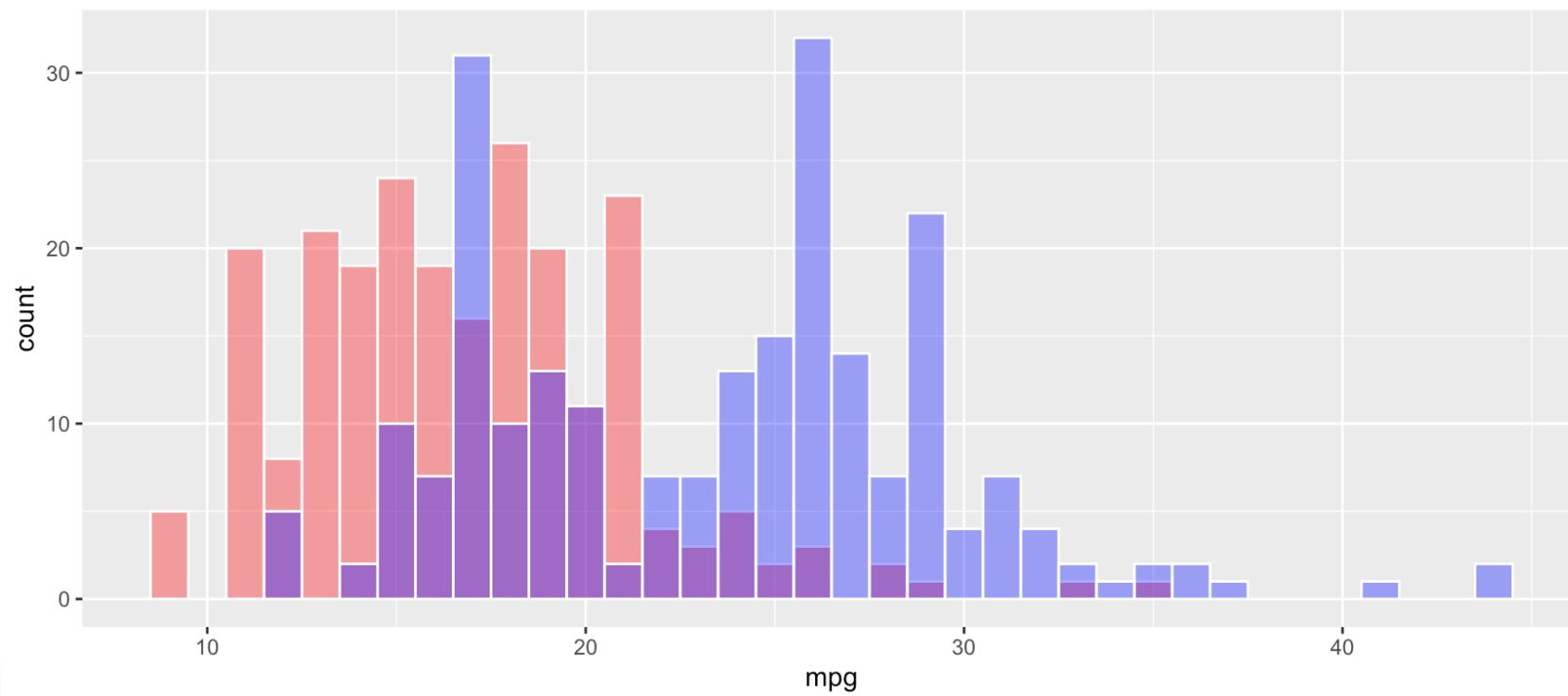
ggplot2 in practice: one variable

```
ggplot(data = mpg, aes(hwy)) +  
  geom_histogram(binwidth = 1, color = "white", fill = "blue", alpha = 0.4)
```



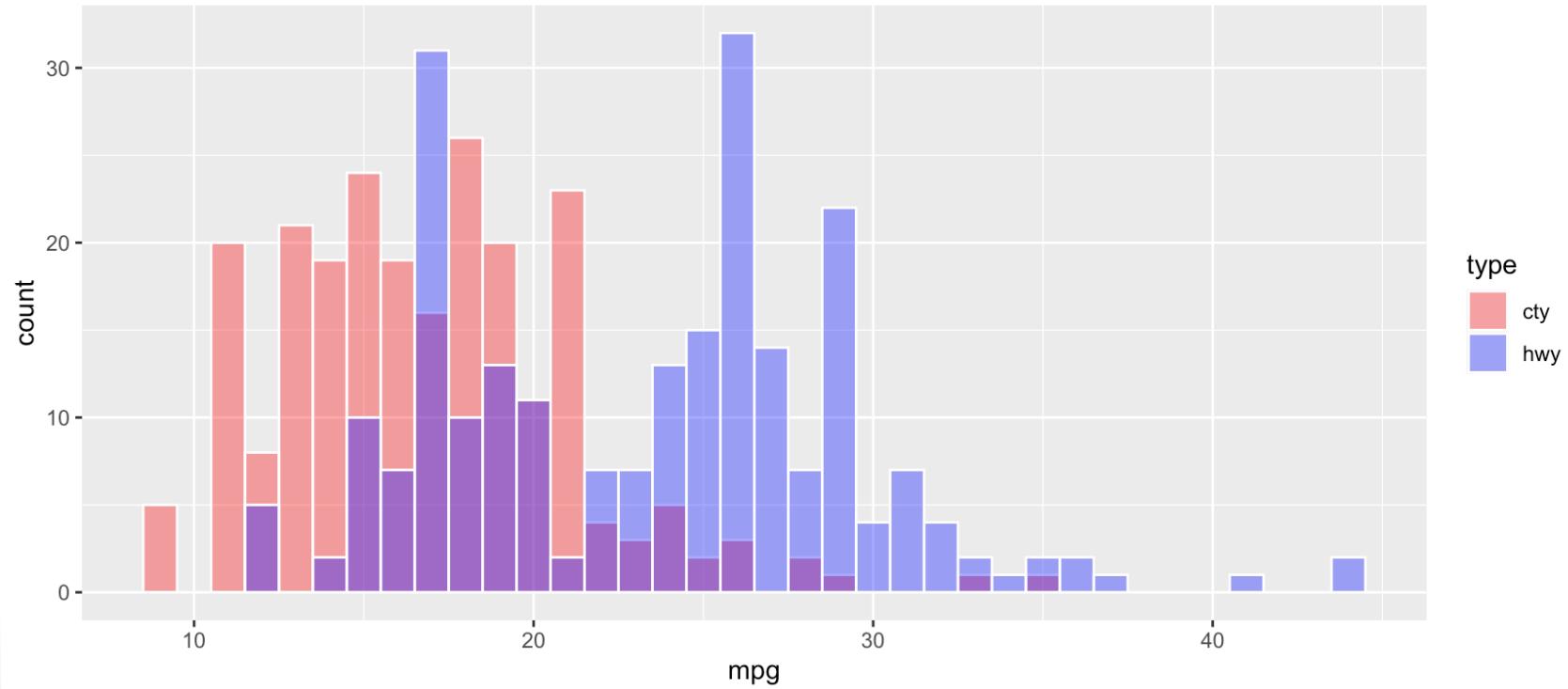
ggplot2 in practice: one variable

```
ggplot(data = mpg) +  
  geom_histogram(aes(cty), binwidth = 1, fill = "red", color = "white", alpha = 0.4) +  
  geom_histogram(aes(hwy), binwidth = 1, fill = "blue", color = "white", alpha = 0.4) +  
  xlab("mpg")
```



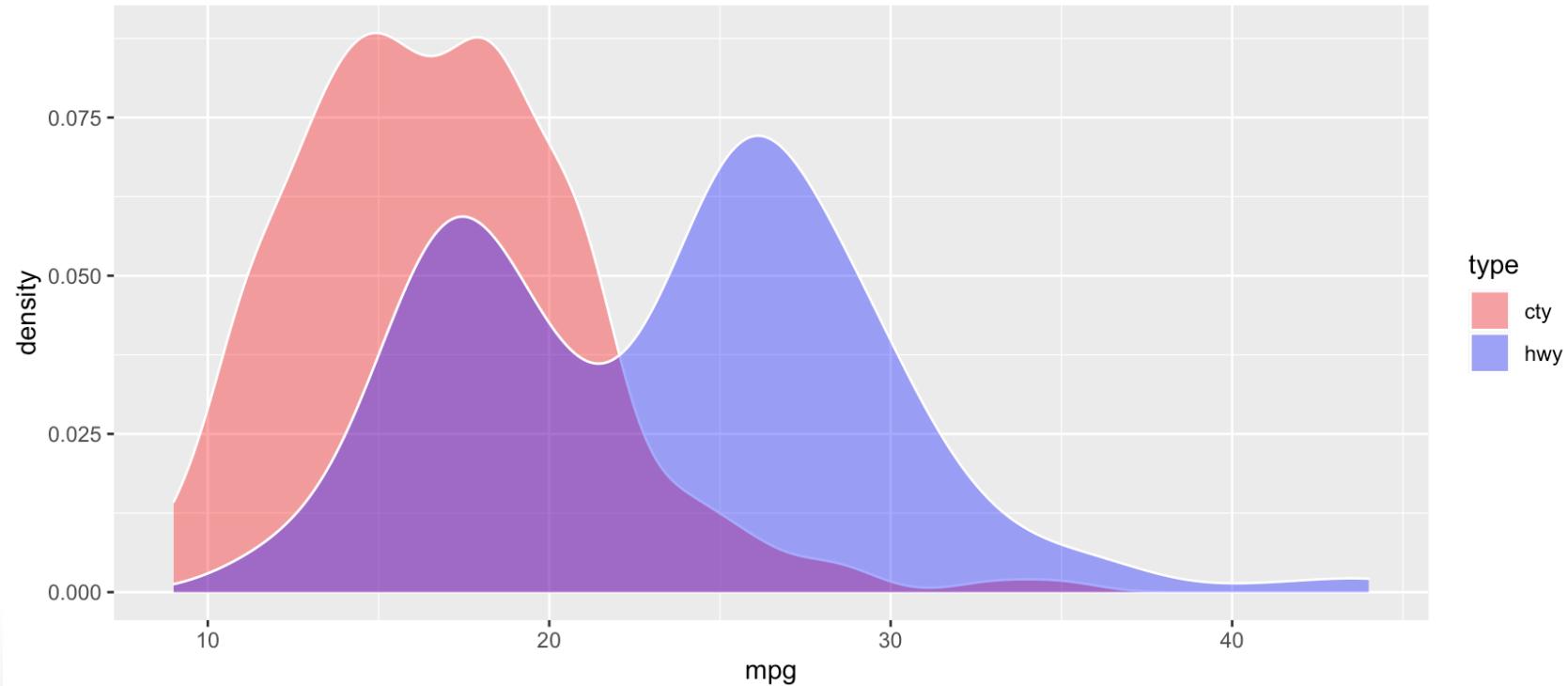
ggplot2 in practice: one variable

```
mpg %>%
  select(model, hwy, cty) %>% pivot_longer(-model, names_to = "type", values_to = "mpg") %>%
  ggplot() +
  geom_histogram(aes(mpg, fill = type), color = "white", position = "identity", binwidth = 1, alpha = 0.4) +
  scale_fill_manual(values=c("red", "blue"))
```



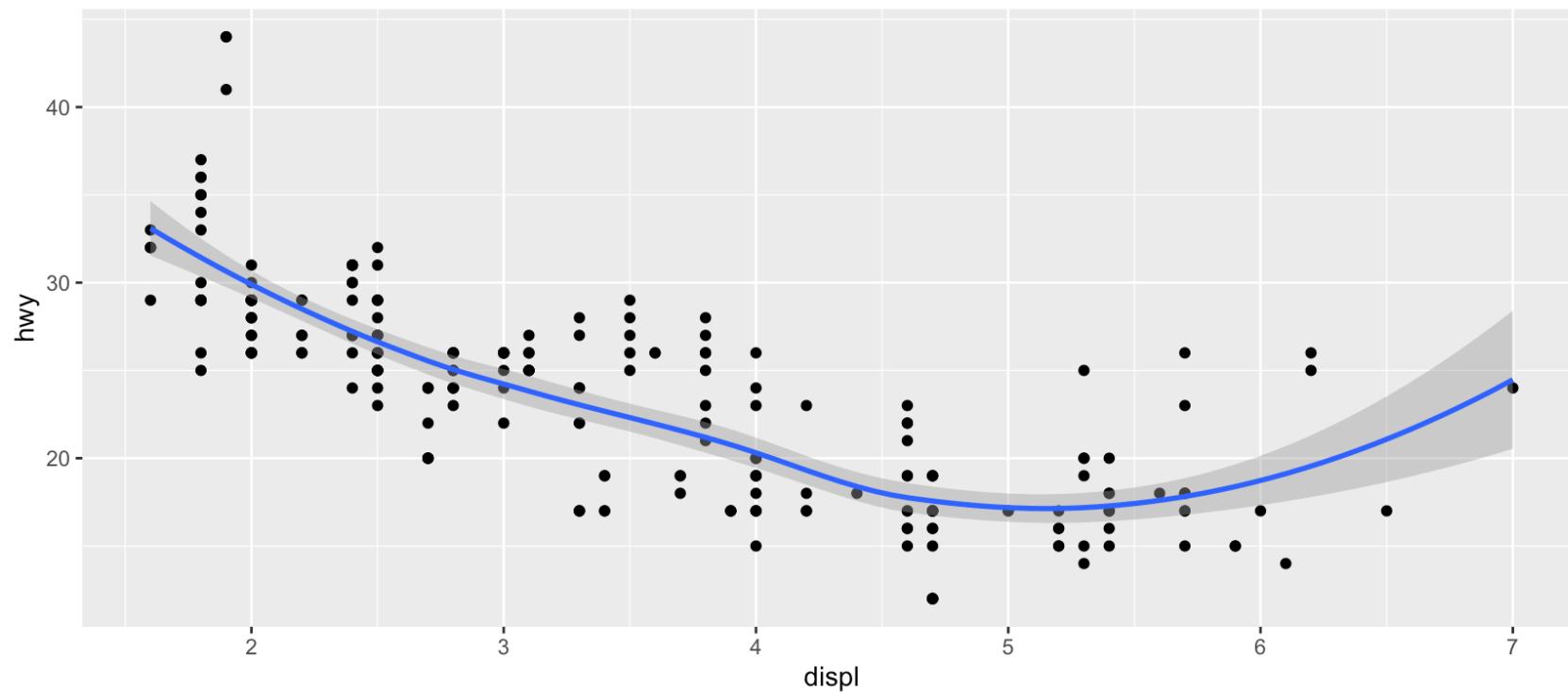
ggplot2 in practice: one variable

```
mpg %>%
  select(model, hwy, cty) %>% pivot_longer(-model, names_to = "type", values_to = "mpg") %>%
  ggplot() +
  geom_density(aes(mpg, fill = type), color = "white", position = "identity", alpha = 0.4) +
  scale_fill_manual(values=c("red", "blue"))
```



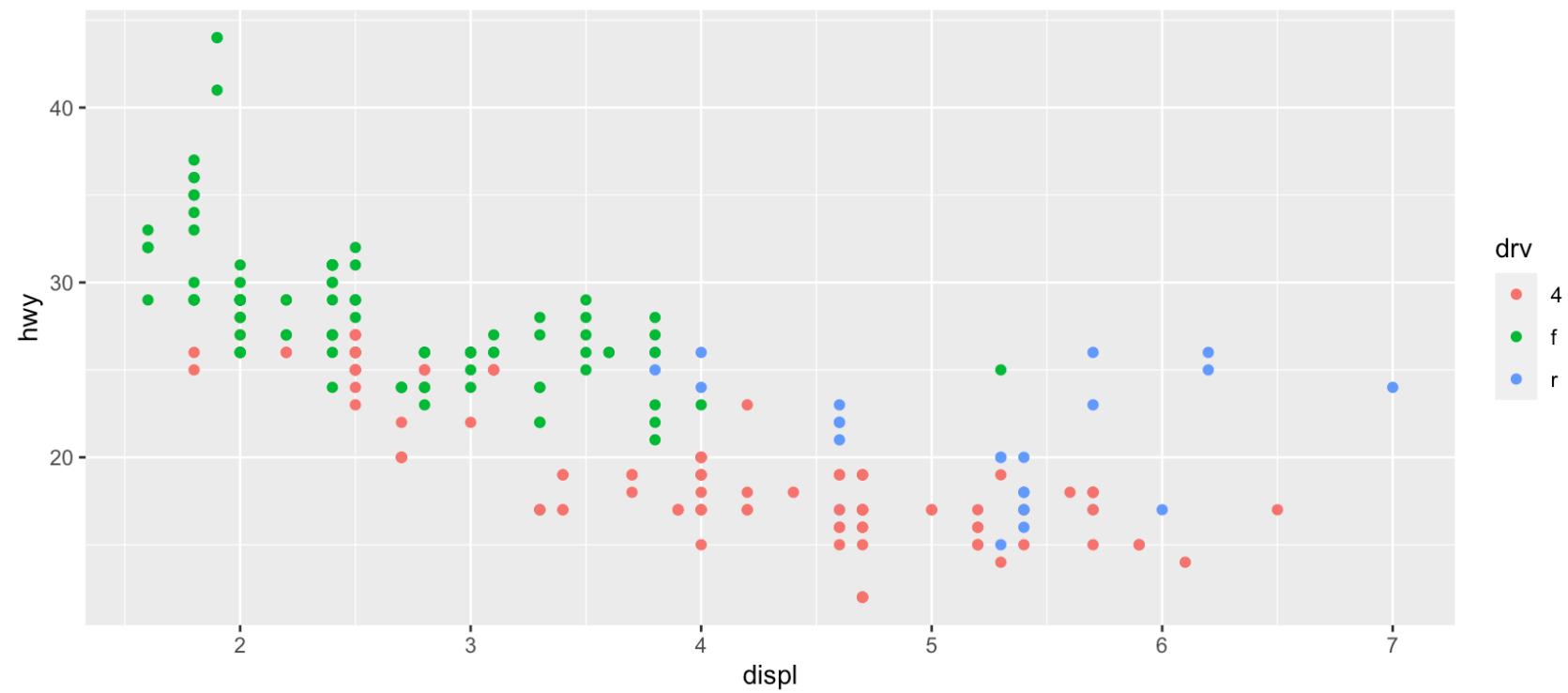
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +  
  geom_point() +  
  geom_smooth()
```



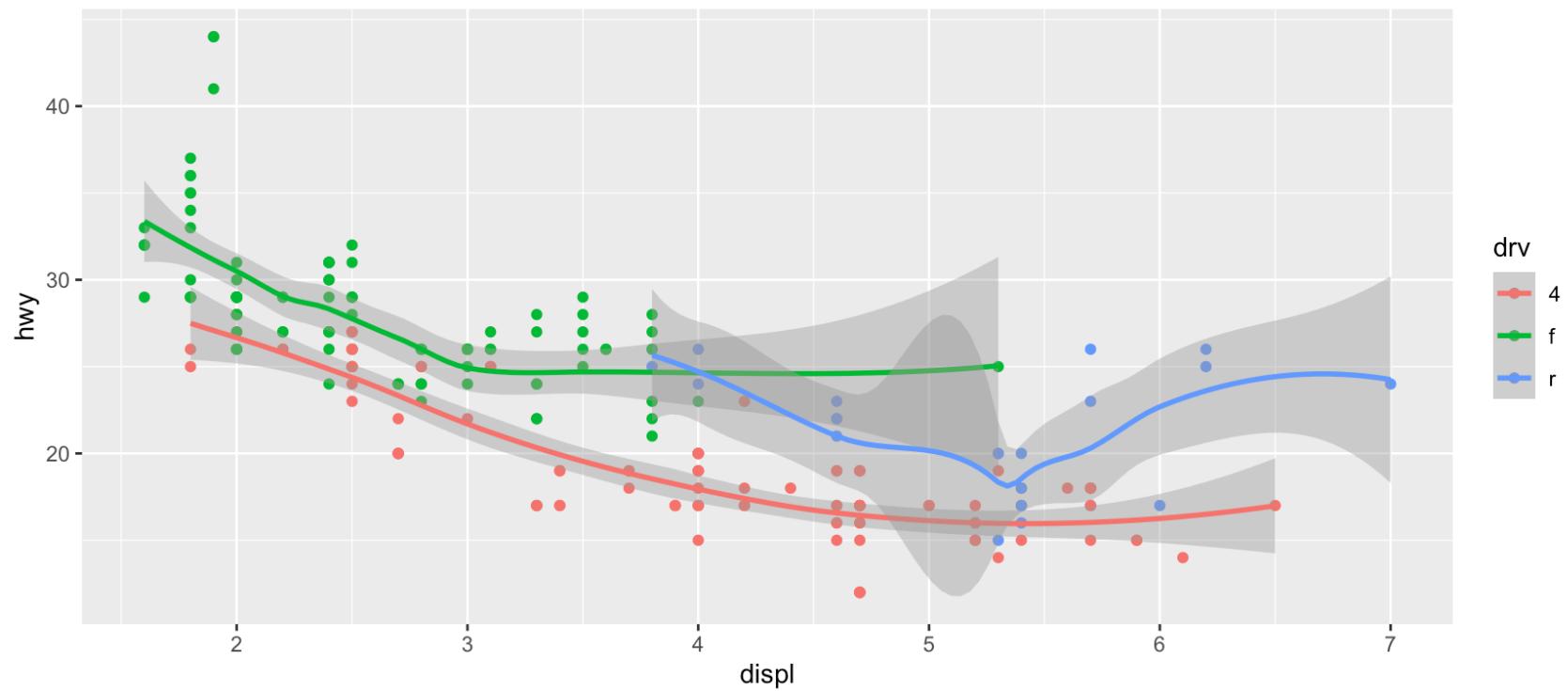
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy, color = drv)) +  
  geom_point()
```



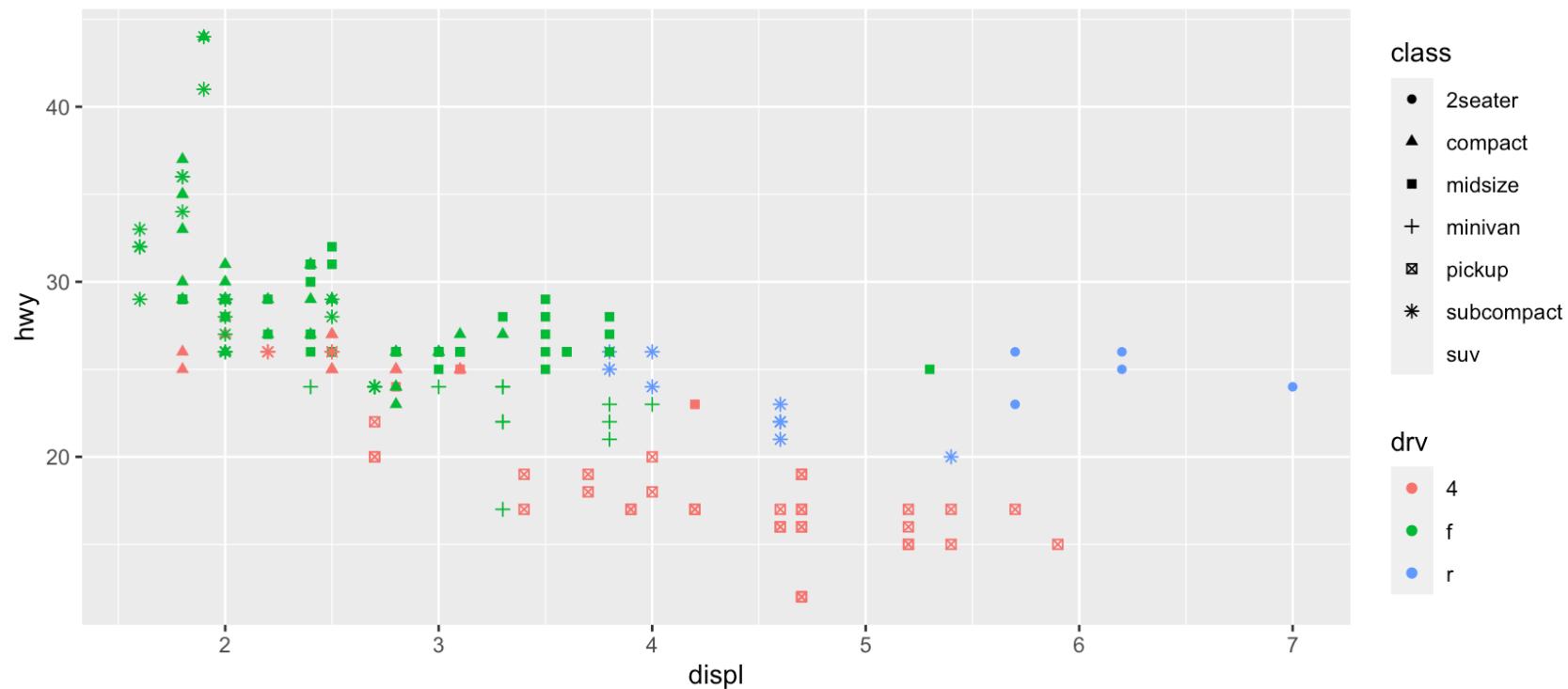
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy, color = drv)) +  
  geom_point() +  
  geom_smooth()
```



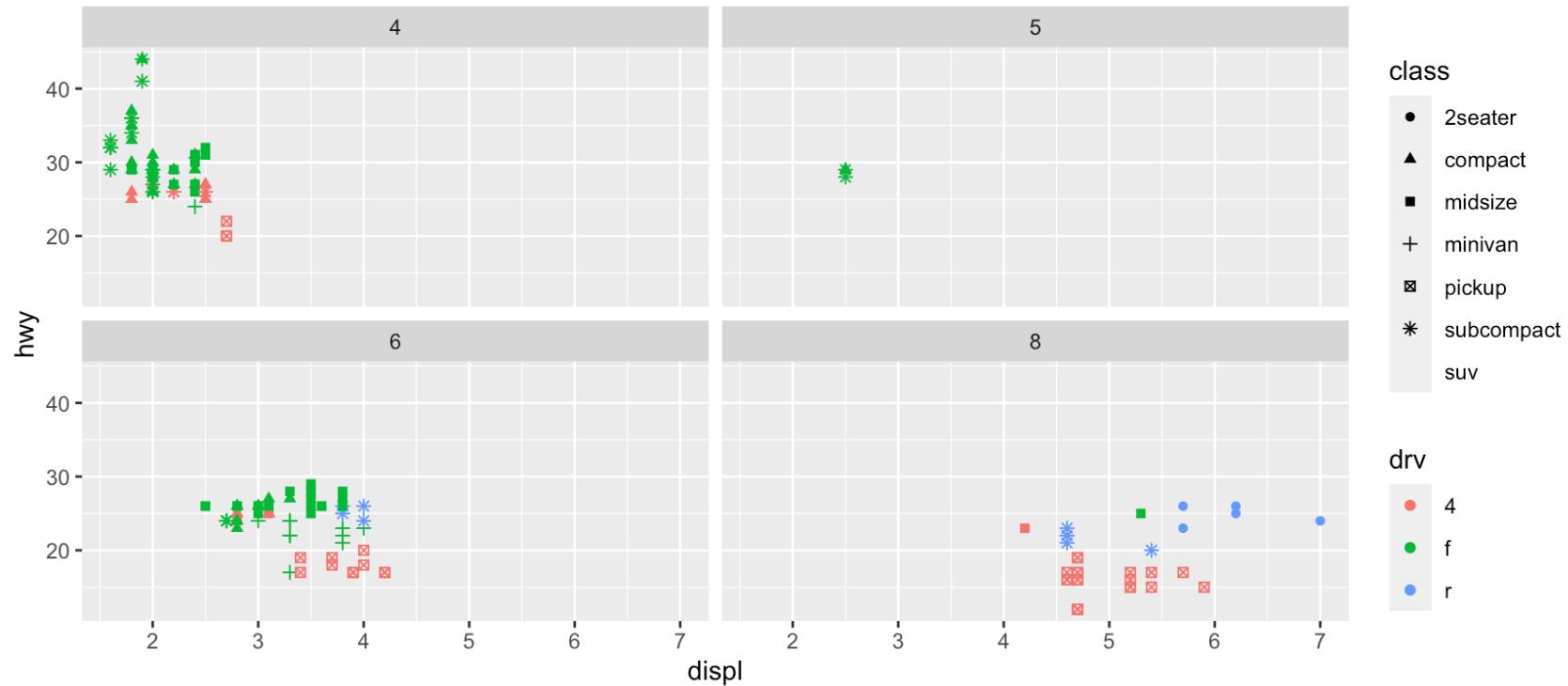
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy, color = drv, shape = class)) +  
  geom_point()
```



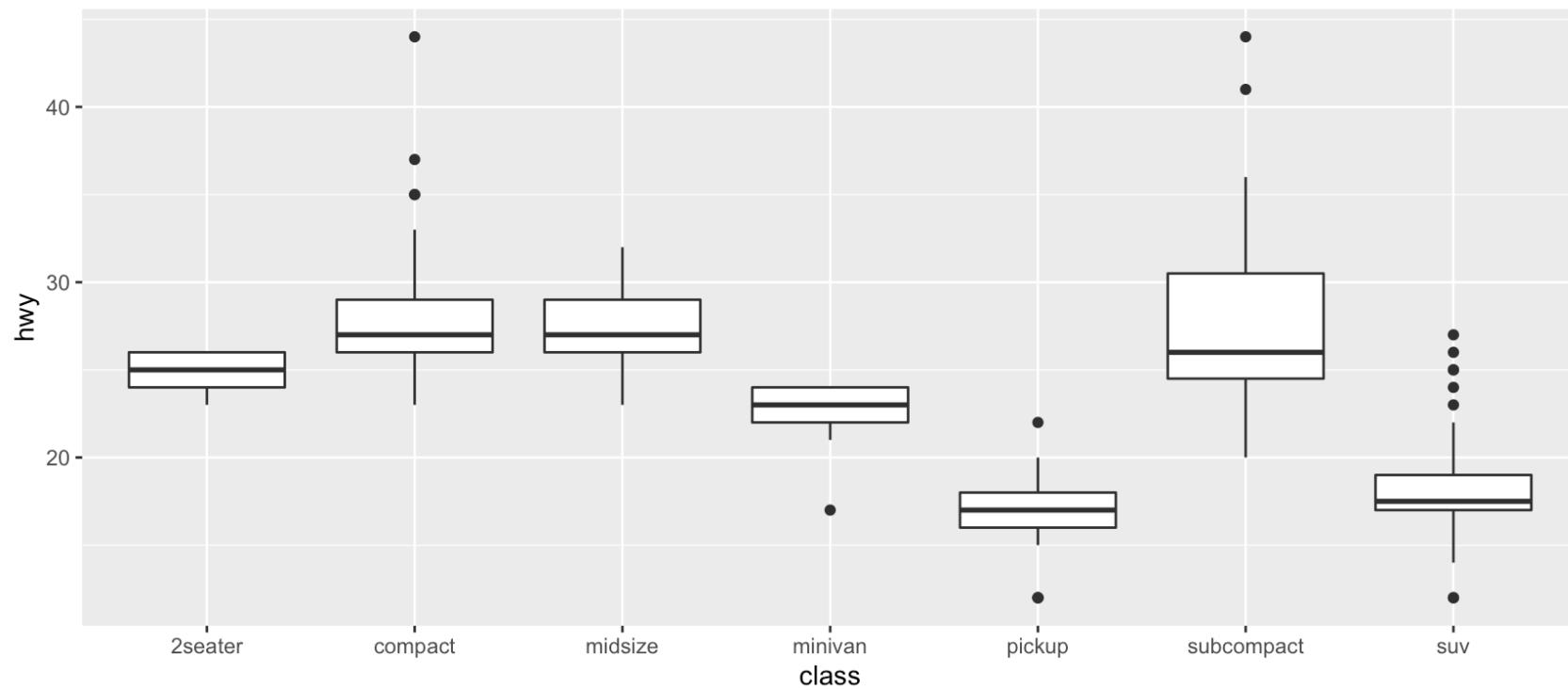
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy, color = drv, shape = class)) +  
  geom_point() +  
  facet_wrap(~cyl)
```



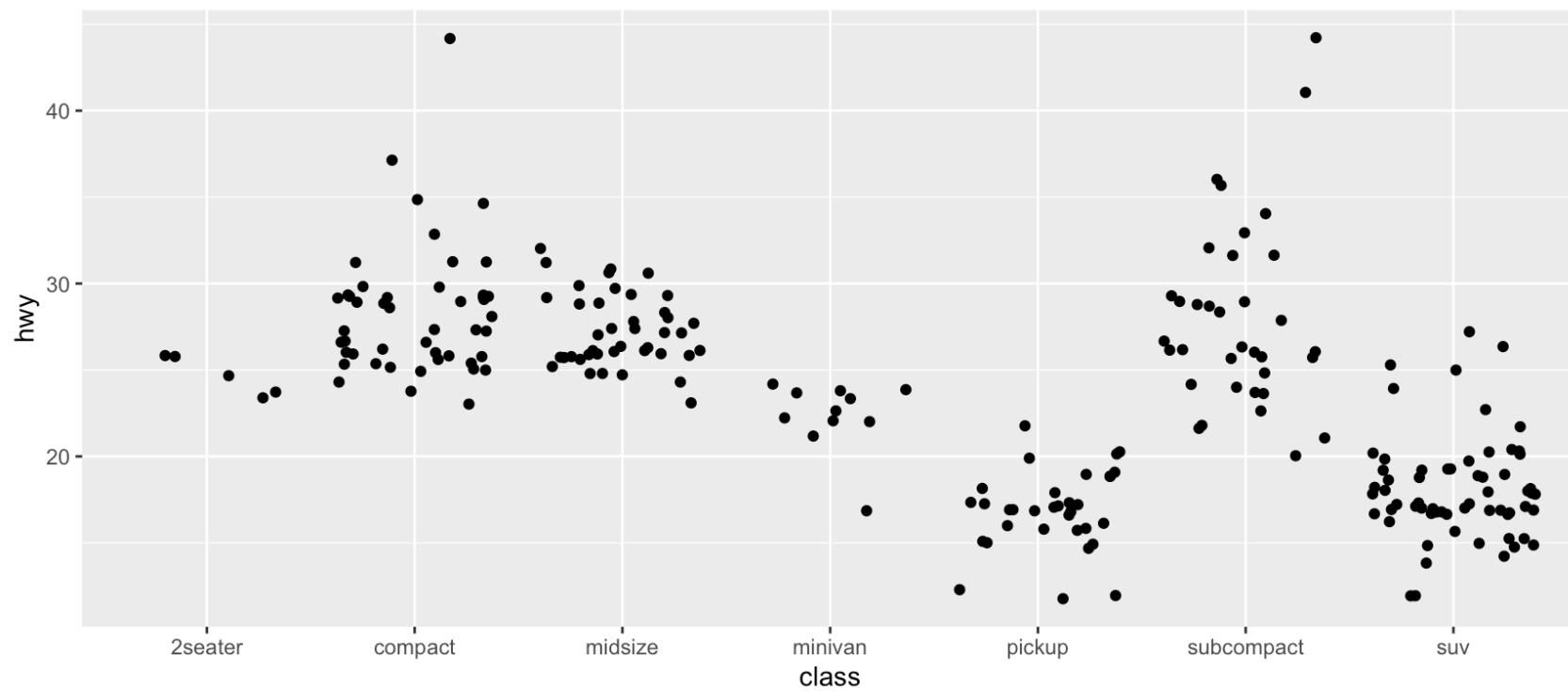
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = class, y = hwy)) +  
  geom_boxplot()
```



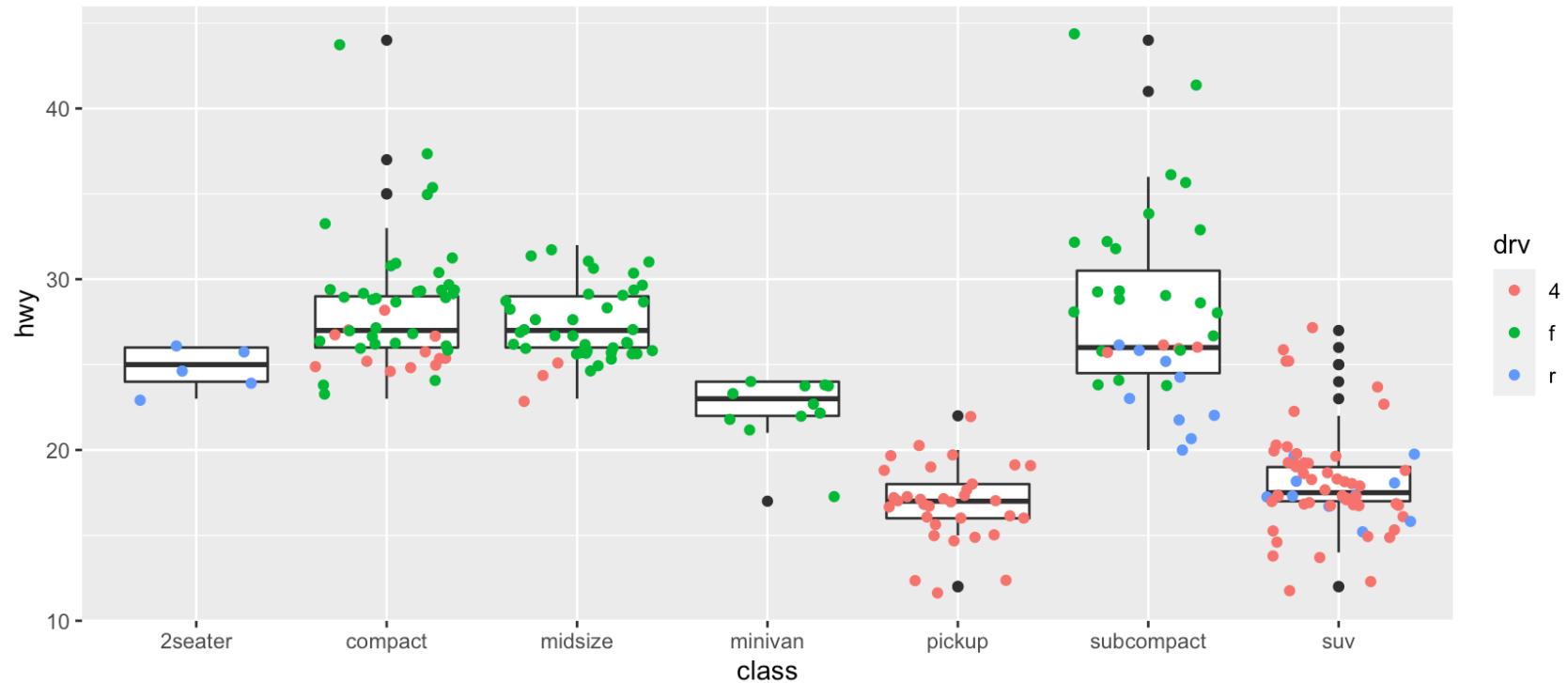
ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = class, y = hwy)) +  
  geom_jitter()
```



ggplot2 in practice: two variables

```
ggplot(data = mpg, aes(x = class, y = hwy)) +  
  geom_boxplot() +  
  geom_jitter(aes(color = drv))
```



ggplot2 in practice: other layers

- `theme()`
- `scale_x_continuous(), scale_y_continuous()`
- `scale_x_log10, scale_y_log10`
- `coord_trans()`
- `annotate()`

ggplot2 and other packages

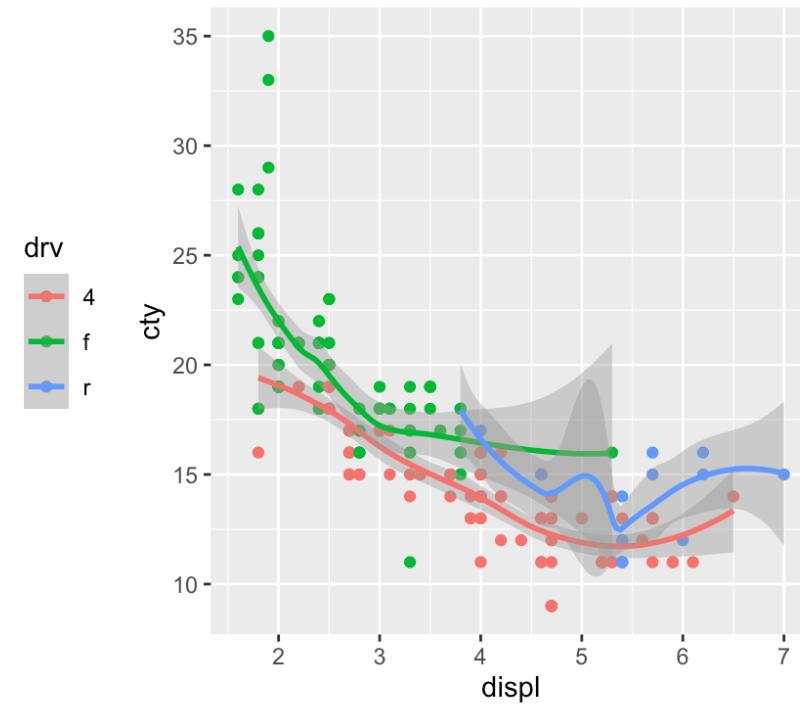
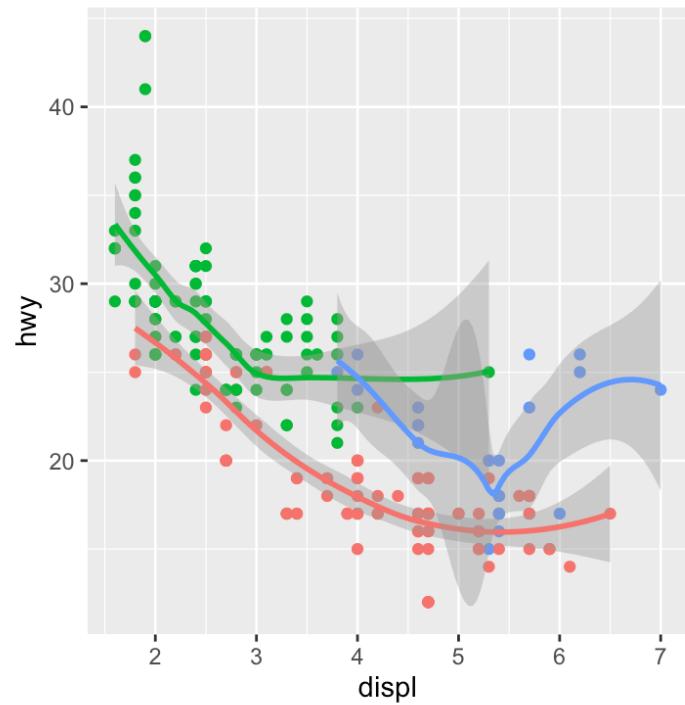
- `patchwork`
- `gghighlight`

ggplot2 in practice: patchwork

```
g_hwy <- ggplot(data = mpg, aes(x = displ, y = hwy, color = drv)) +  
  geom_point() +  
  geom_smooth()  
  
g_cty <- ggplot(data = mpg, aes(x = displ, y = cty, color = drv)) +  
  geom_point() +  
  geom_smooth()
```

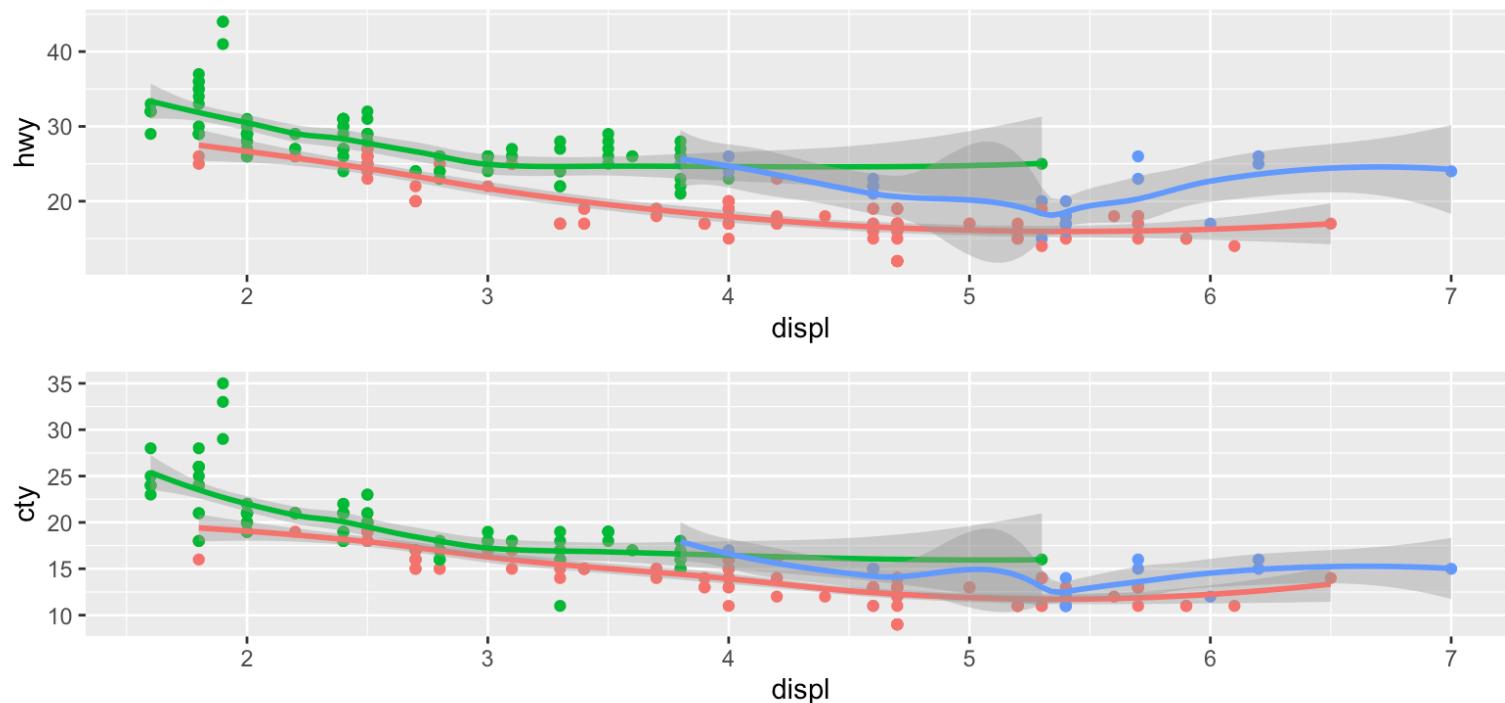
ggplot2 in practice: patchwork

```
library(patchwork)
g_hwy + g_cty
```



ggplot2 in practice: patchwork

```
library(patchwork)
g_hwy / g_cty
```

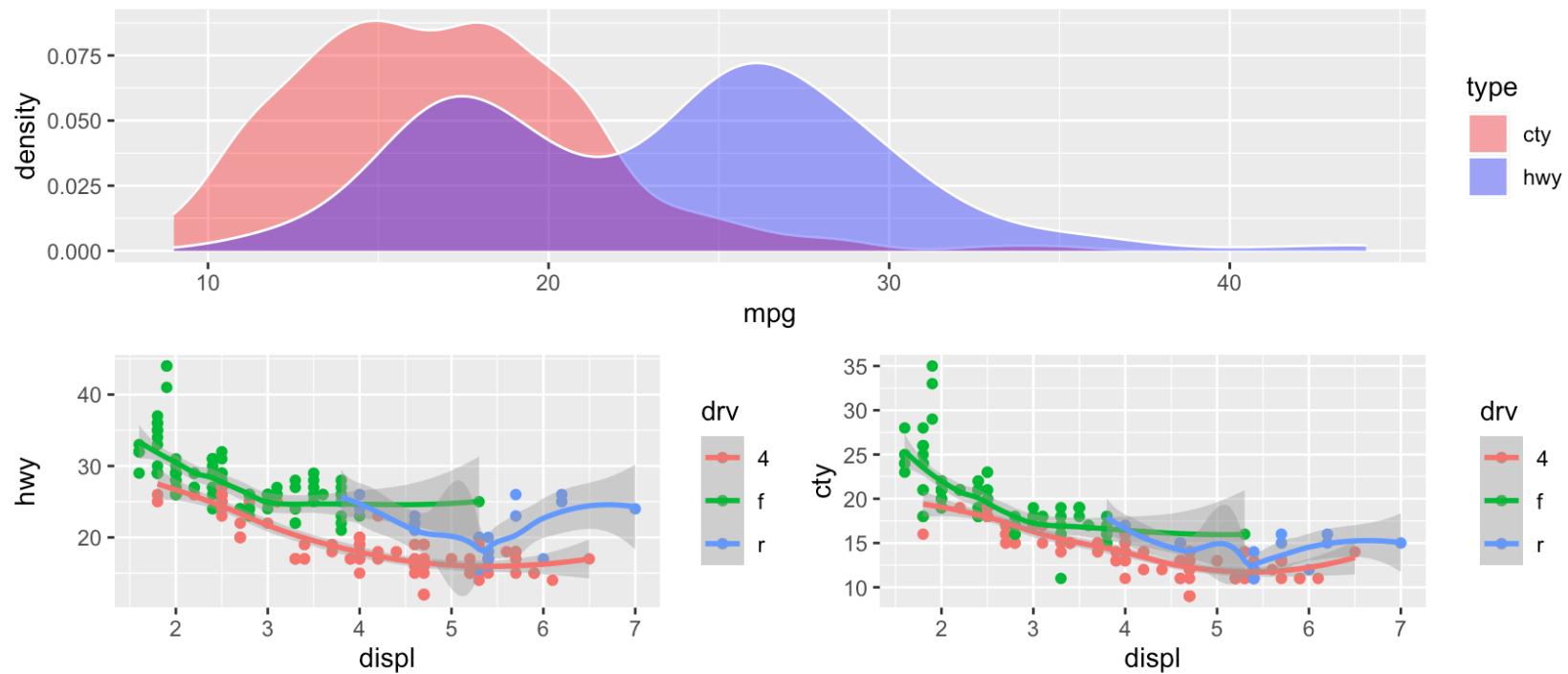


ggplot2 in practice: patchwork

```
g_den <- mpg %>%
  select(-model, -hwy, -cty) %>% pivot_longer(-model, names_to = "type", values_to = "mpg") %>%
  ggplot() +
  geom_density(aes(mpg, fill = type), color = "white", position = "identity", alpha = 0.4) +
  scale_fill_manual(values=c("red", "blue"))
```

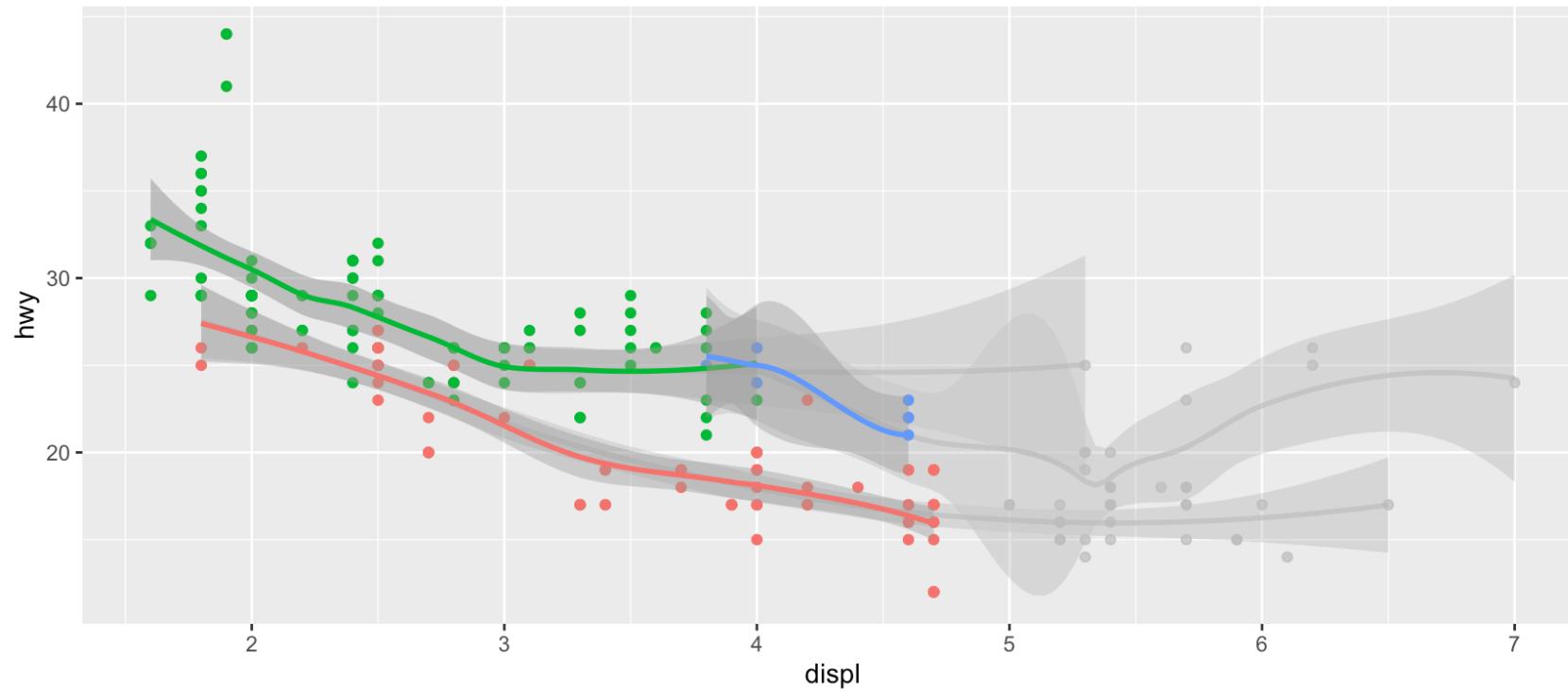
ggplot2 in practice: patchwork

```
g_den / (g_hwy + g_cty)
```



ggplot2 in practice: gghighlight

```
library(gghighlight)
g_hwy +
  gghighlight(displ < 5)
```



Questions

References

References

- Bone, Frederique, and Daniele Rotolo. 2020. "Text mining historical sources to trace technological change. The case of mass production." Working Paper.
- Börner, Katy. 2016. *Atlas of Science*. Cambridge, Massachusetts: The MIT Press.
- Camerani, Roberto, Daniele Rotolo, and Nicola Grassano. 2018. "Do firms publish? A Multi-Sectoral Analysis." *SPRU Working Paper Series* 21: 1–38. <https://www.sussex.ac.uk/webteam/gateway/file.php?name=2018-21-swps-camerani-et-al.pdf&site=25>.
- Harris, Robert L. 1996. *Information Graphics: A Comprehensive Illustrated Reference*. Oxford; New York: Oxford University Press.
- Hausmann, Ricardo, César A Hidalgo, Sebastián Bustos, Michele Coscia, Alexander Simoes, and Muhammed A Yıldırım. 2013. *The Atlas of Economic Complexity*. MIT Press.
<http://www.jstor.org/stable/j.ctt9qf8jp>
- Jackman, Robert W. 1980. "The Impact of Outliers on Income Inequality." *American Sociological Review* 45 (2): 344–47. <https://doi.org/10.2307/2095134>.
- Tufte, Edward R. 2001. *The Visual Display of Quantitative Information*. Vol. 4. Cheshire, Connecticut: Graphic Press.
- VanderMeer, Jeff. 2013. *Wonderbook: The Illustrated Guide to Creating Imaginative Fiction*. New York, NY, USA: Abrams Image.