

Mapping Data to Graphics

Session 3

PMAP 8551/4551: Data Visualization with R
Andrew Young School of Policy Studies
Spring 2026

Plan for today

Data, aesthetics, & the grammar of graphics

Grammatical layers

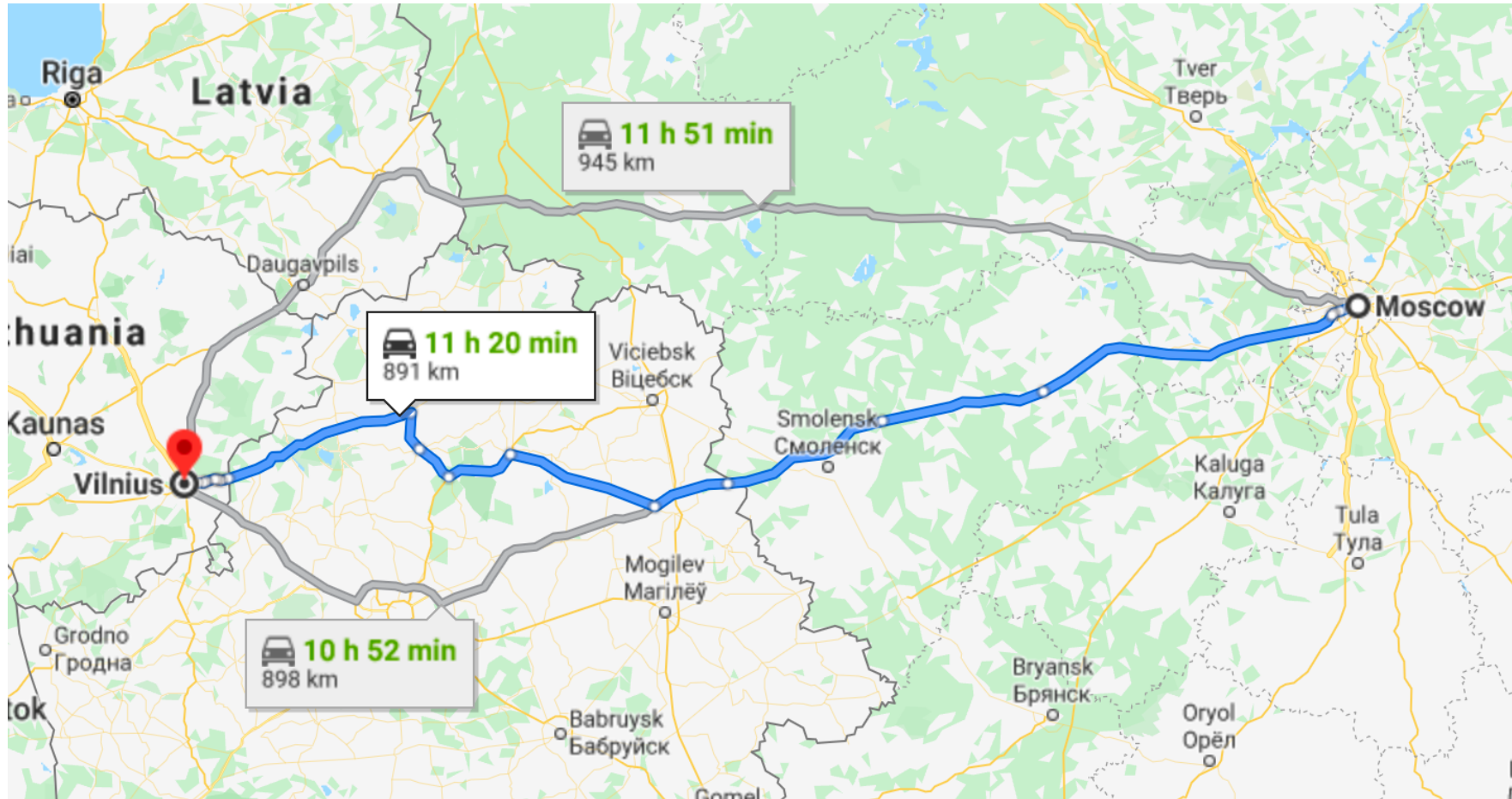
Aesthetics in extra dimensions

Tidy data

Data, aesthetics, & the grammar of graphics

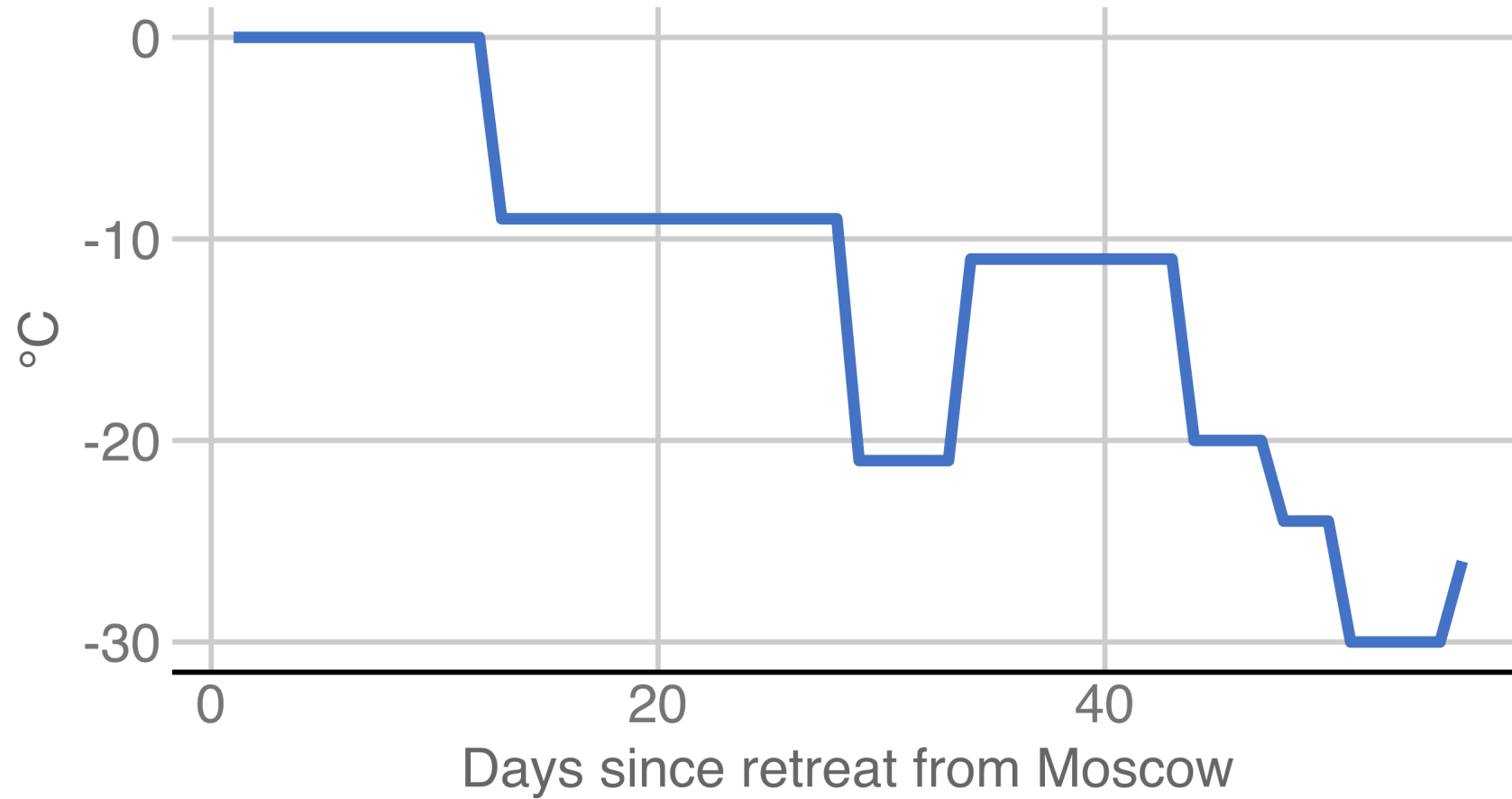


Long distance!



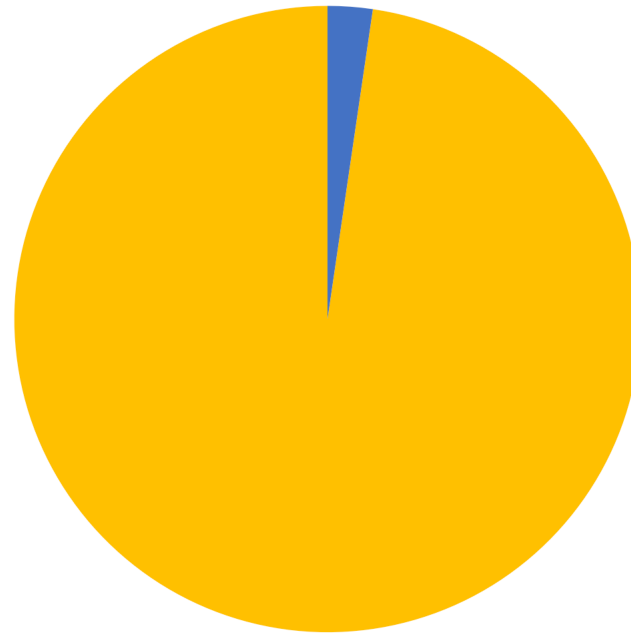
Moscow to Vilnius

Very cold!



Lots of people died!

Napoleon's Grande Armée



■ Died ■ Survived

Carte Figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie 1812-1813.

Dressée par M. Minard, Inspecteur Général des Ponts et Chaussées en retraite. Paris, le 20 Novembre 1869.

Les nombres d'hommes présents sont représentés par les largeurs des zones colorées à raison d'un millimètre pour dix mille hommes; ils sont de plus écrits en travers des zones. Le rouge désigne les hommes qui entrent en Russie; le noir ceux qui en sortent. Les renseignements qui ont servi à dresser la carte ont été puisés dans les ouvrages de M. M. Chiers, de Ségur, de Fezensac, de Chambray et le journal inédit de Jacob, pharmacien de l'Armée depuis le 28 Octobre.

Pour mieux faire juger à l'œil la diminution de l'armée, j'ai supposé que les corps du Prince Jérôme et du Maréchal Davout qui avaient été détachés sur Minsk et Mohilow et qui rejoignent vers Orscha et Witebsk, avaient toujours marché avec l'armée.

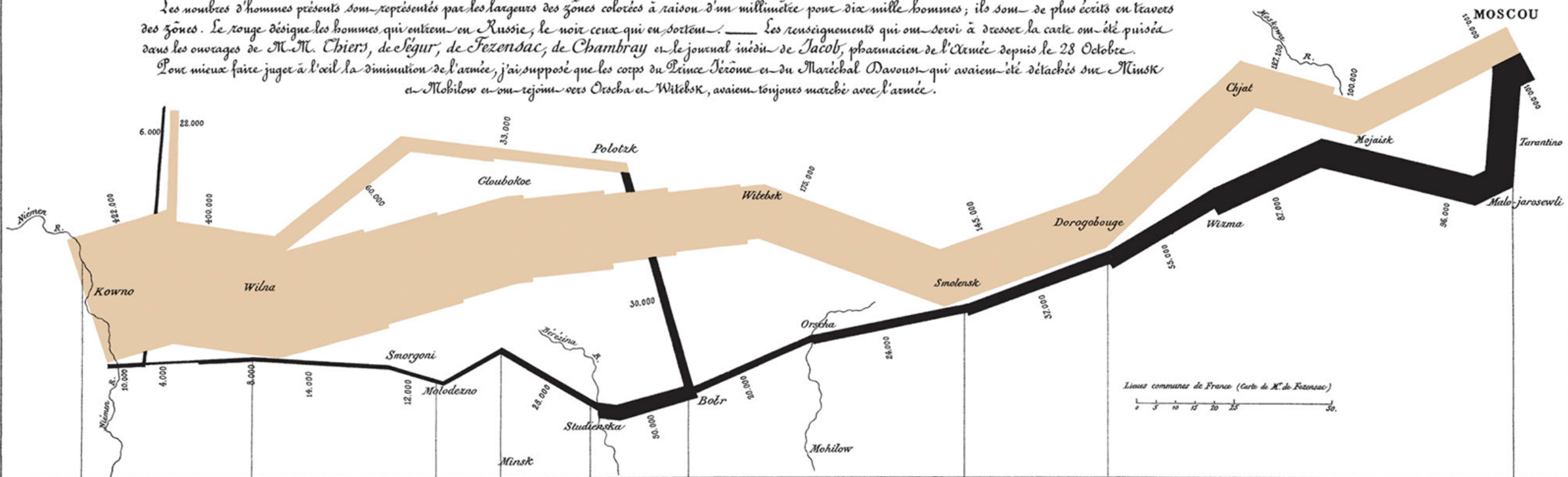
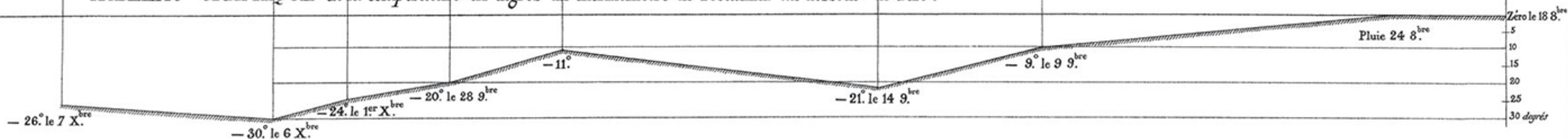
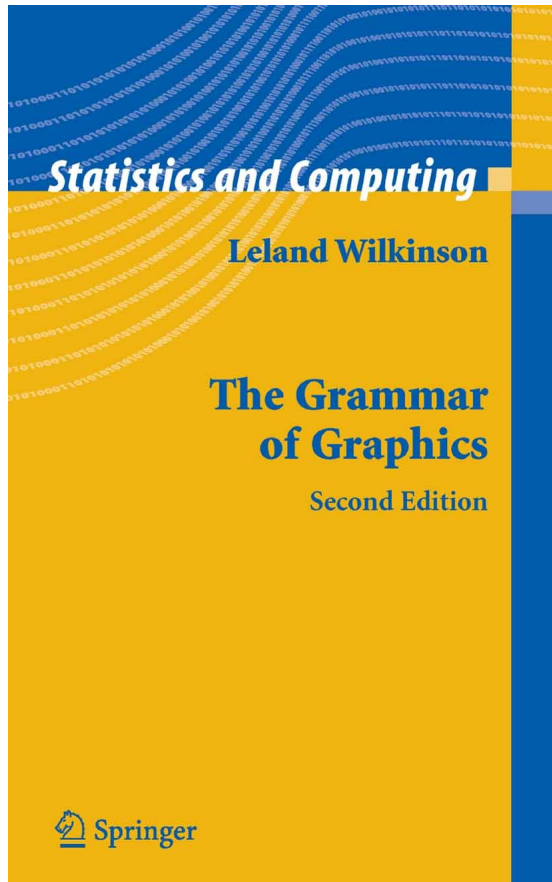


TABLEAU GRAPHIQUE de la température en degrés du thermomètre de Réaumur au dessous de zéro.



Les Cosaques passent au galop le Niemen gelé.

Mapping data to aesthetics



Aesthetic

Visual property of a graph

Position, shape, color, etc.

Data

A column in a dataset

Mapping data to aesthetics

Data	Aesthetic	Graphic/Geometry
Longitude	Position (x-axis)	Point
Latitude	Position (y-axis)	Point
Army size	Size	Path
Army direction	Color	Path
Date	Position (x-axis)	Line + text
Temperature	Position (y-axis)	Line + text

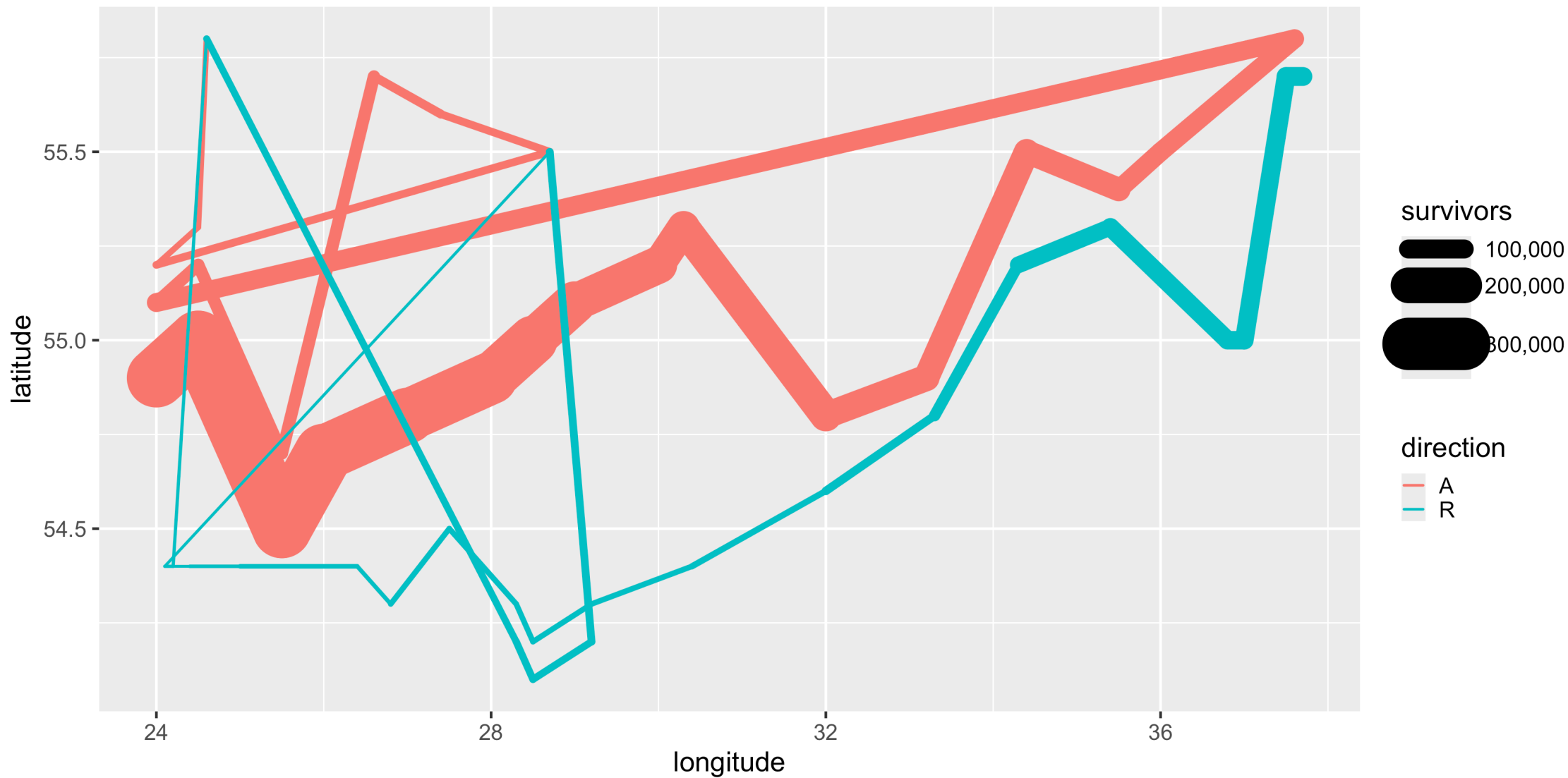
Mapping data to aesthetics

Data	aes()	geom
Longitude	x	geom_point()
Latitude	y	geom_point()
Army size	size	geom_path()
Army direction	color	geom_path()
Date	x	geom_line() + geom_text()
Temperature	y	geom_line() + geom_text()

ggplot() template

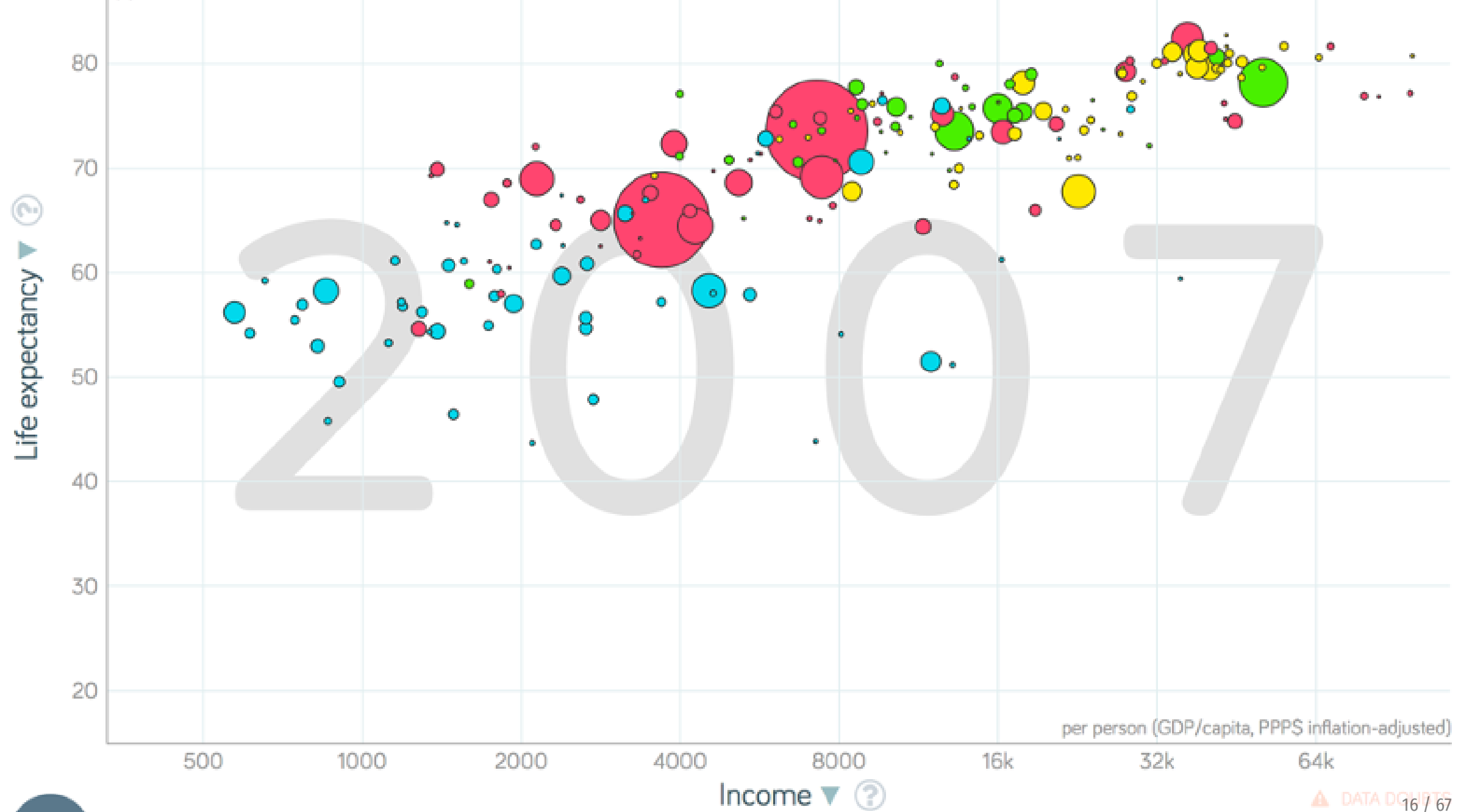
This is a dataset named `troops`:

longitude	latitude	direction	survivors
24	54.9	A	340000
24.5	55	A	340000
...	...	...	...





ntly organized TED event



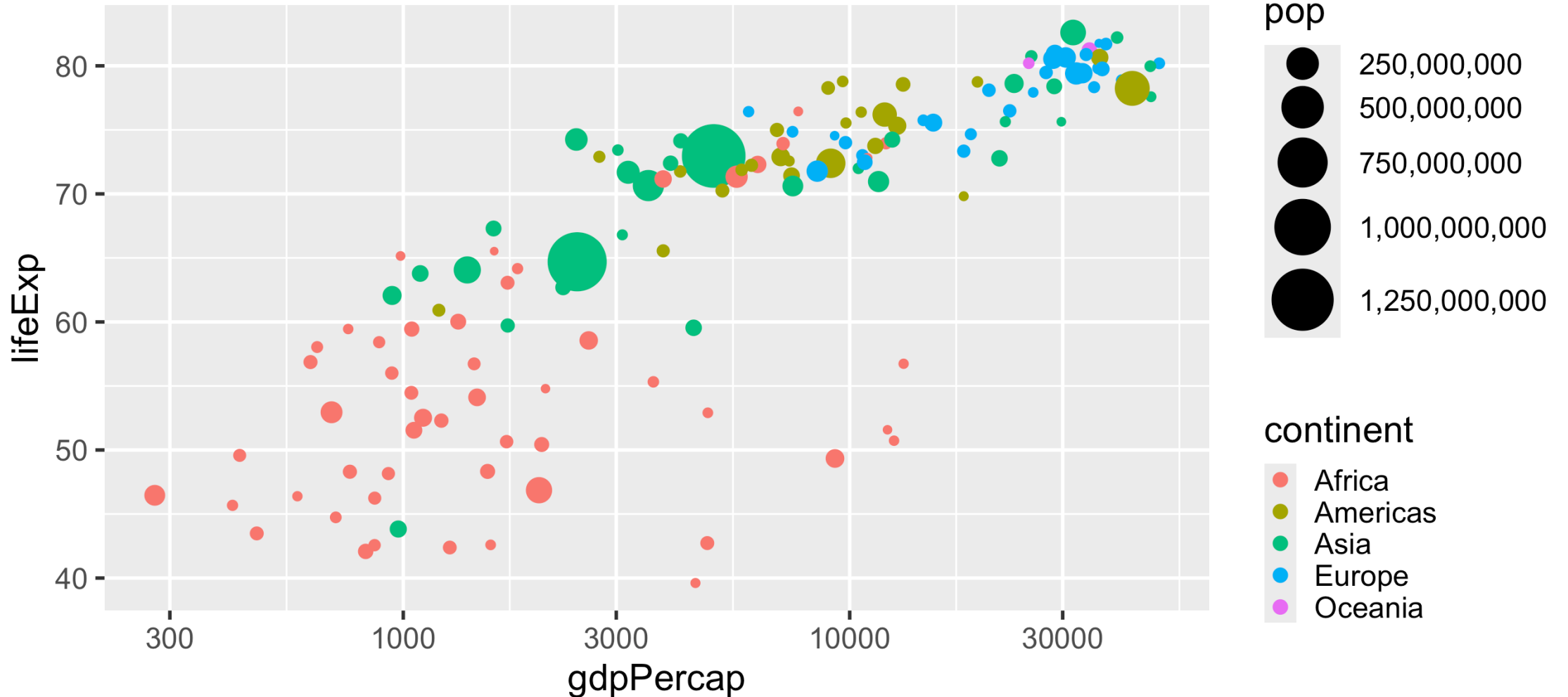
Mapping data to aesthetics

Data	aes()	geom
Wealth (GDP/capita)	x	geom_point()
Health (Life expectancy)	y	geom_point()
Continent	color	geom_point()
Population	size	geom_point()

This is a dataset named `gapminder_2007`:

country	continent	gdpPercap	lifeExp	pop
Afghanistan	Asia	974.5803384	43.828	31889923
Albania	Europe	5937.029526	76.423	3600523
...	...	...	...	...

Health and wealth



Grammatical layers

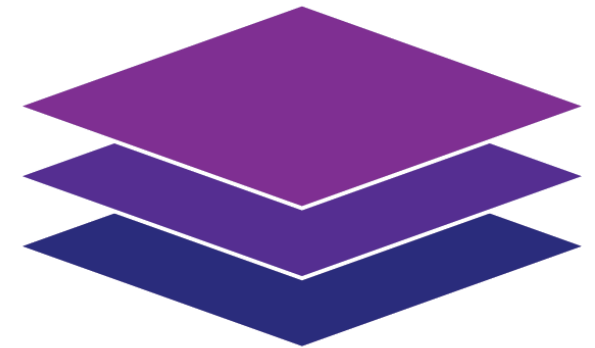
Grammar components as layers

So far we know about data,
aesthetics, and geometries

Think of these
components as **layers**

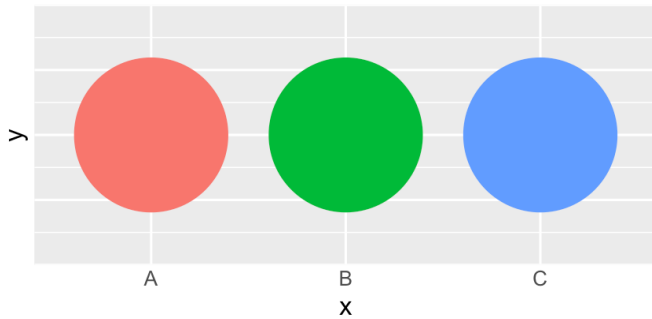
Add them to foundational
`ggplot()` with **+**

Geometries
Aesthetics
Data

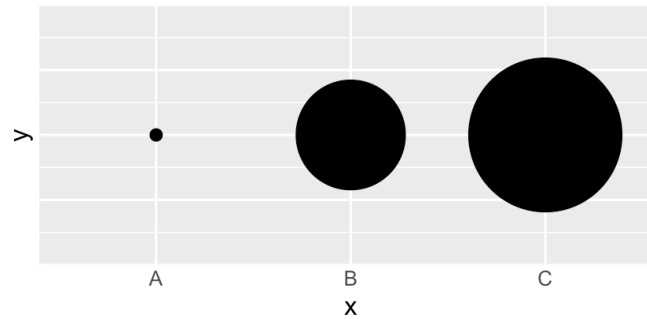


Possible aesthetics

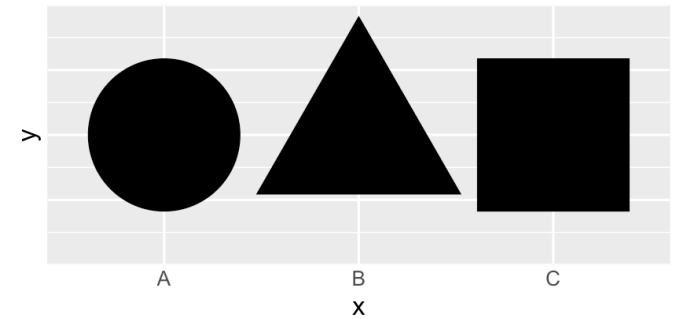
color (discrete)



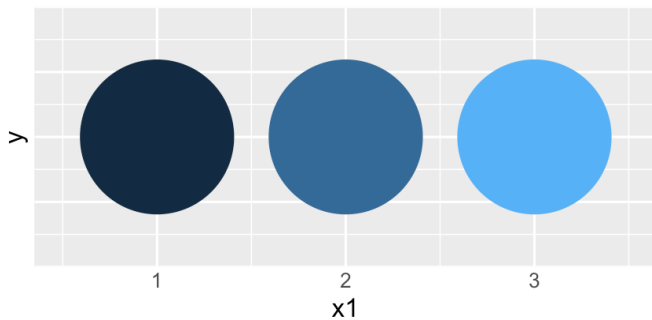
size



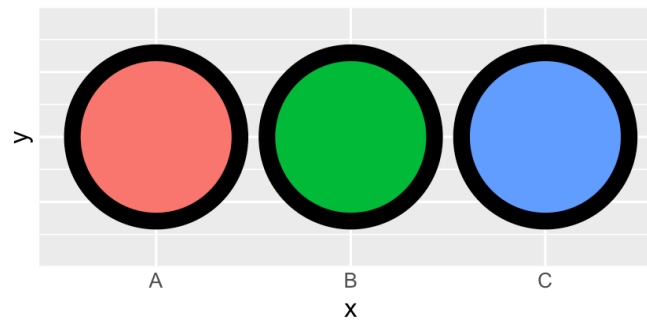
shape



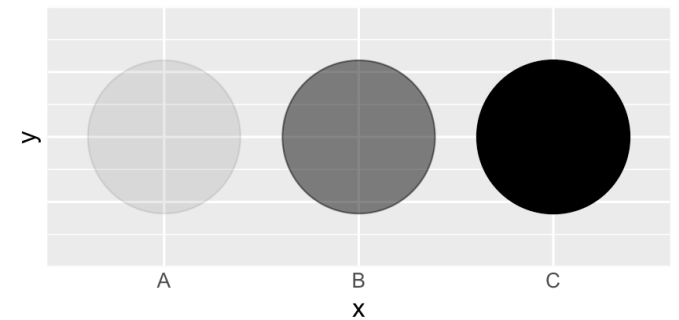
color (continuous)



fill



alpha



Possible geoms

	Example geom	What it makes
	<code>geom_col()</code>	Bar charts
	<code>geom_text()</code>	Text
	<code>geom_point()</code>	Points
	<code>geom_boxplot()</code>	Boxplots
	<code>geom_sf()</code>	Maps

Possible geoms

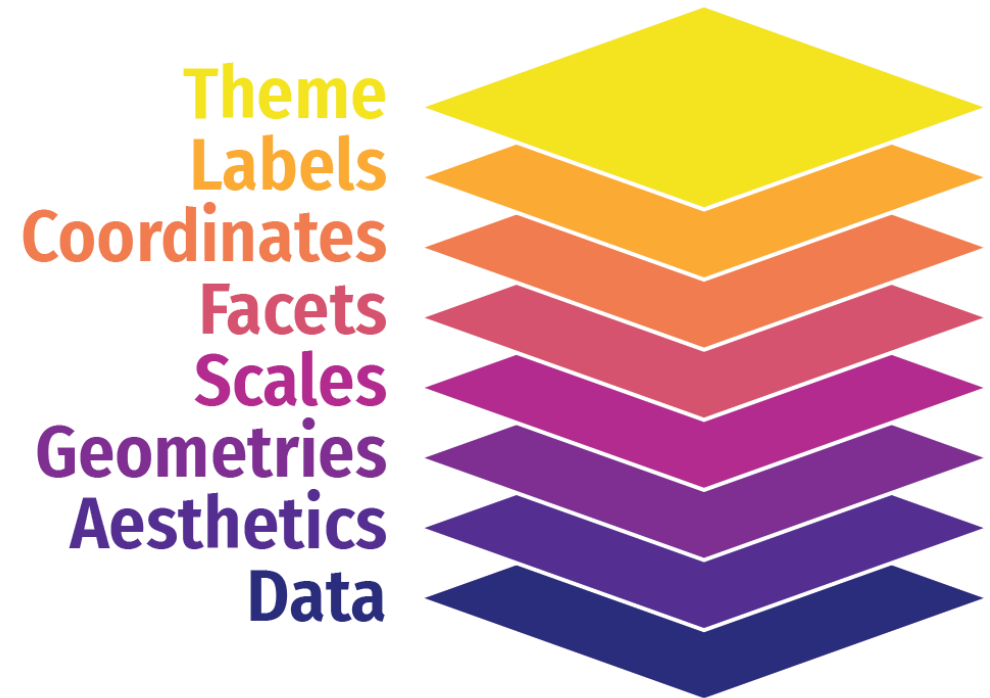
There are dozens of possible geoms and each class session will cover different ones.

See **the `{ggplot2}` documentation** for complete examples of all the different geom layers

Additional layers

There are many of other grammatical layers we can use to describe graphs!

We sequentially add layers onto the foundational `ggplot()` plot to create complex figures



Scales

Scales change the properties of the variable mapping

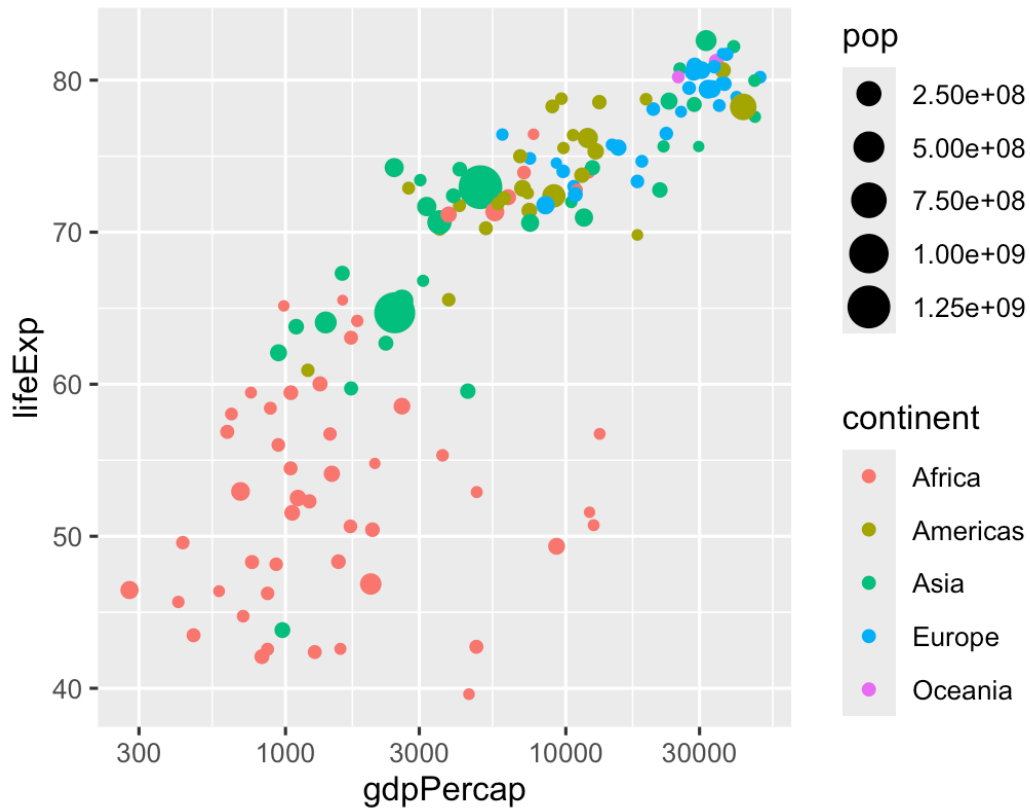
Example layer

What it does

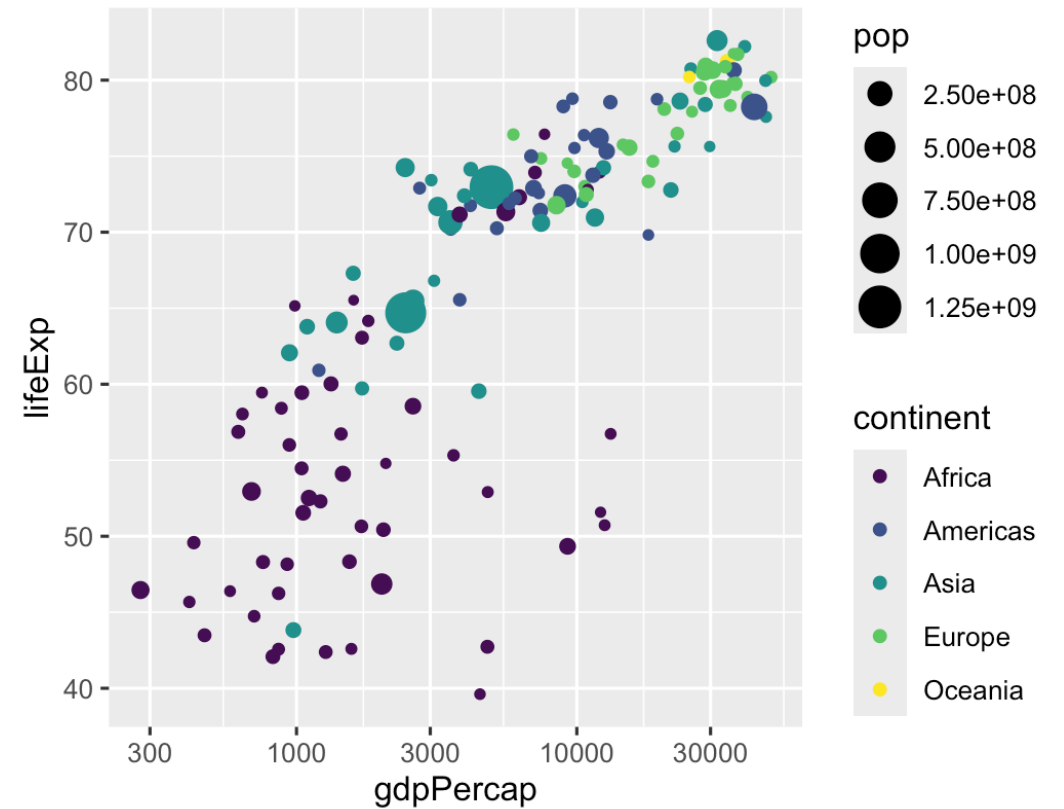
<code>scale_x_continuous()</code>	Make the x-axis continuous
<code>scale_x_continuous(breaks = 1:5)</code>	Manually specify axis ticks
<code>scale_x_log10()</code>	Log the x-axis
<code>scale_color_gradient()</code>	Use a gradient
<code>scale_fill_viridis_d()</code>	Fill with discrete viridis colors

Scales

`scale_x_log10()`



`scale_color_viridis_d()`



Facets

Facets show subplots for different subsets of data

Example layer

What it does

```
facet_wrap(vars(continent))
```

Plot for each continent

```
facet_wrap(vars(continent, year))
```

Plot for each continent/year

```
facet_wrap(..., ncol = 1)
```

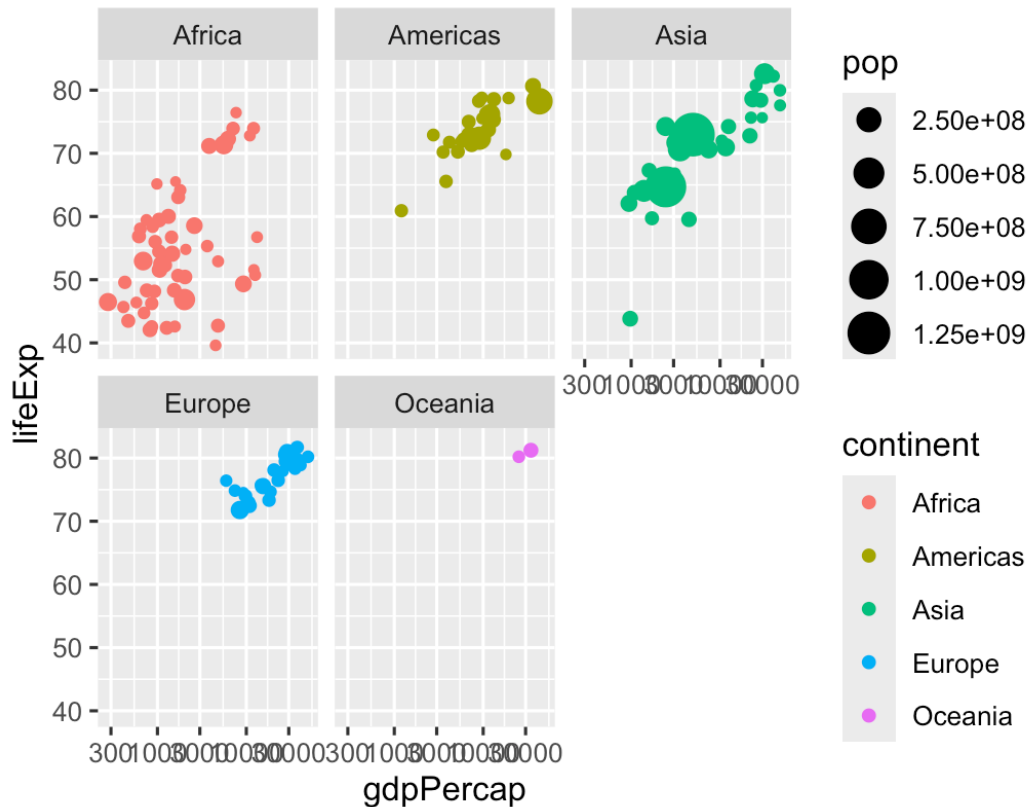
Put all facets in one column

```
facet_wrap(..., nrow = 1)
```

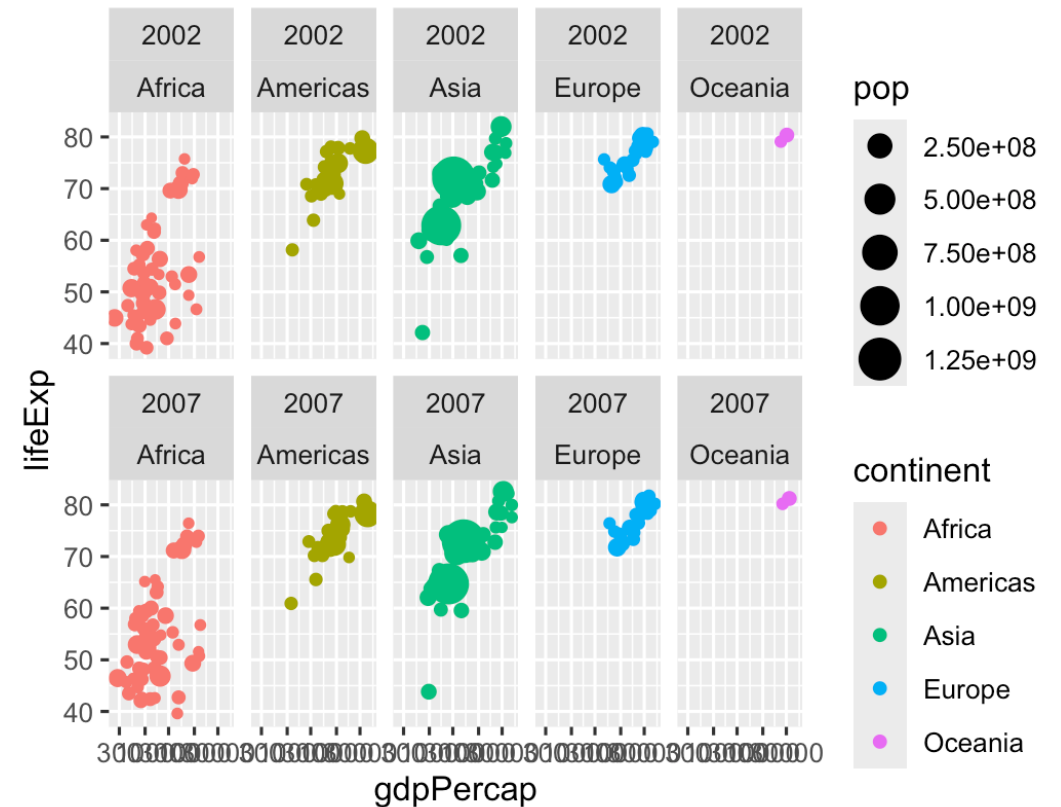
Put all facets in one row

Facets

```
facet_wrap(vars(continent))
```



```
facet_wrap(vars(continent, year))
```



Coordinates

Change the coordinate system

Example layer

```
coord_cartesian()
```

```
coord_cartesian(ylim = c(1, 10))
```

```
coord_flip()
```

```
coord_polar()
```

What it does

Plot for each continent

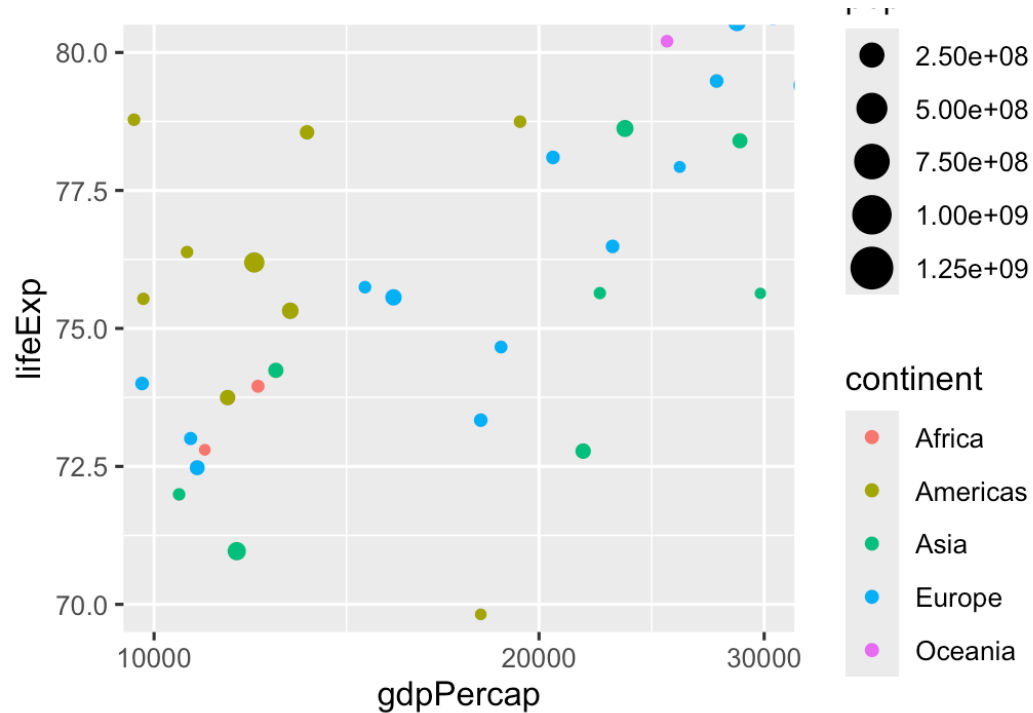
Zoom in where y is 1–10

Switch x and y

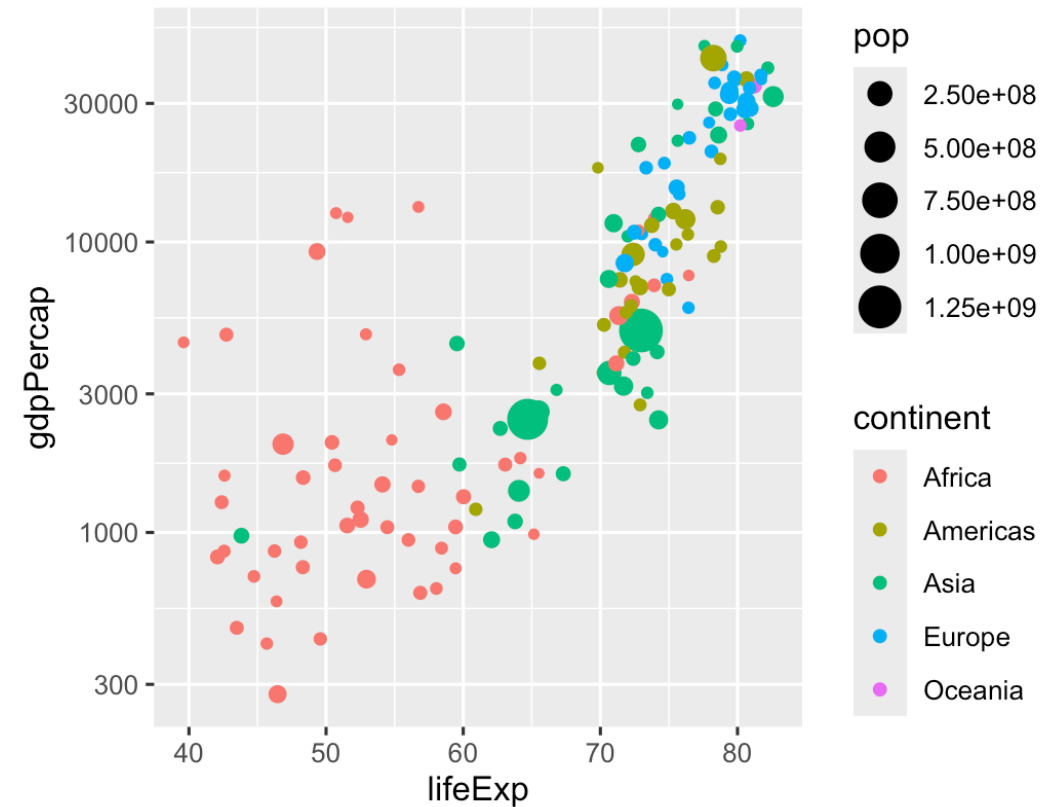
Use circular polar system

Coordinates

```
coord_cartesian(ylim = c(70, 80),  
               xlim = c(10000, 30000))
```



```
coord_flip()
```



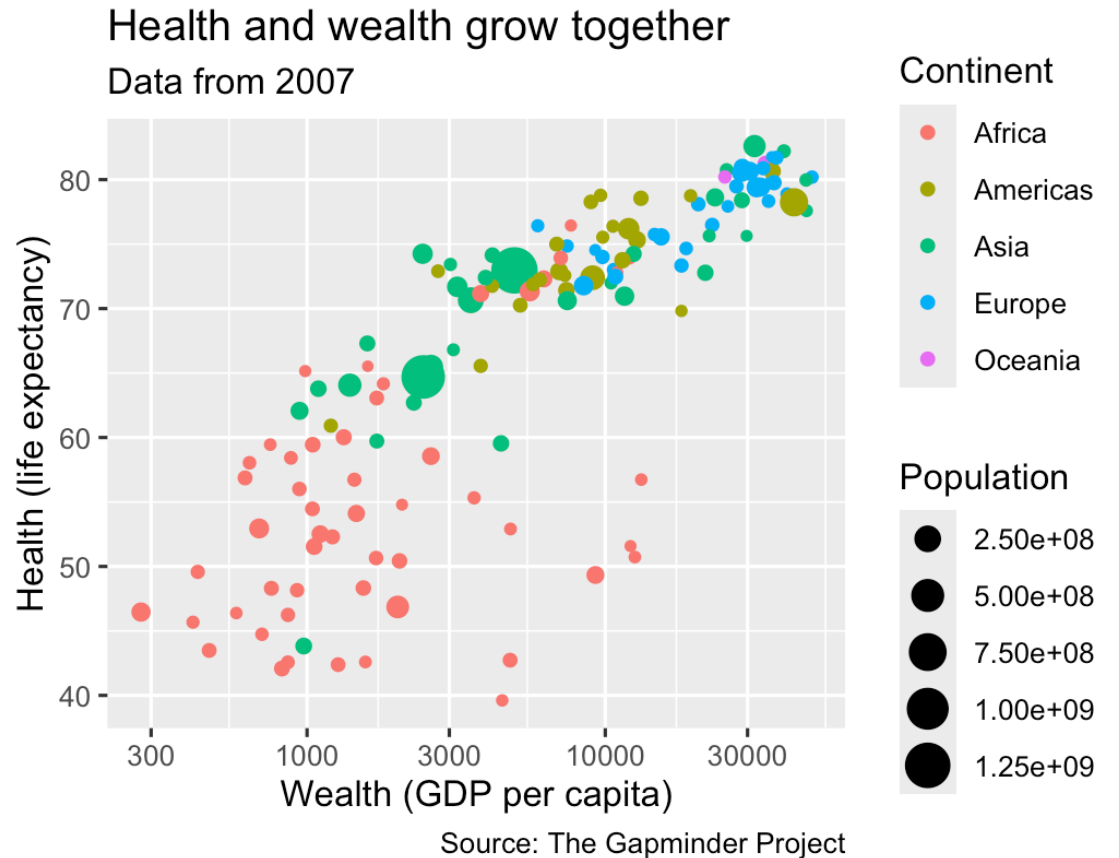
Labels

Add labels to the plot with a single `labs()` layer

Example layer	What it does
<code>labs(title = "Neat title")</code>	Title
<code>labs(caption = "Something")</code>	Caption
<code>labs(y = "Something")</code>	y-axis
<code>labs(size = "Population")</code>	Title of size legend

Labels

```
ggplot(gapminder_2007,  
  aes(x = gdpPercap, y = lifeExp,  
      color = continent, size = pop)) +  
  geom_point() +  
  scale_x_log10() +  
  labs(title = "Health and wealth grow together",  
       subtitle = "Data from 2007",  
       x = "Wealth (GDP per capita)",  
       y = "Health (life expectancy)",  
       color = "Continent",  
       size = "Population",  
       caption = "Source: The Gapminder Project")
```



Theme

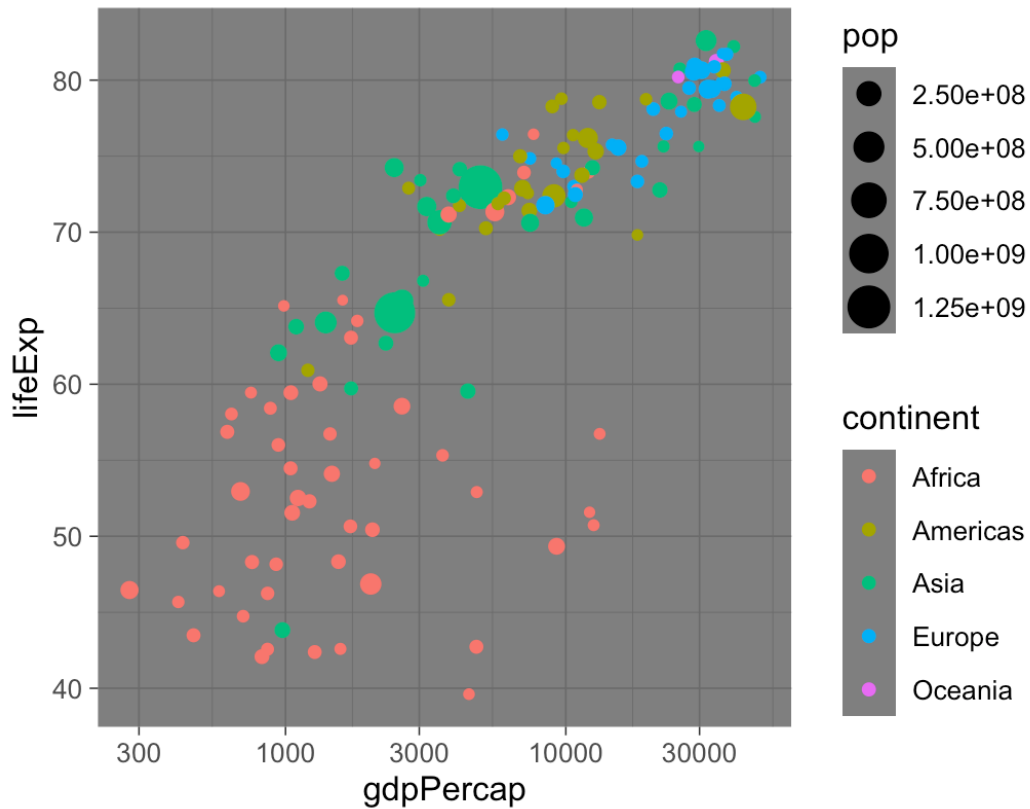
Change the appearance of anything in the plot

There are many built-in themes

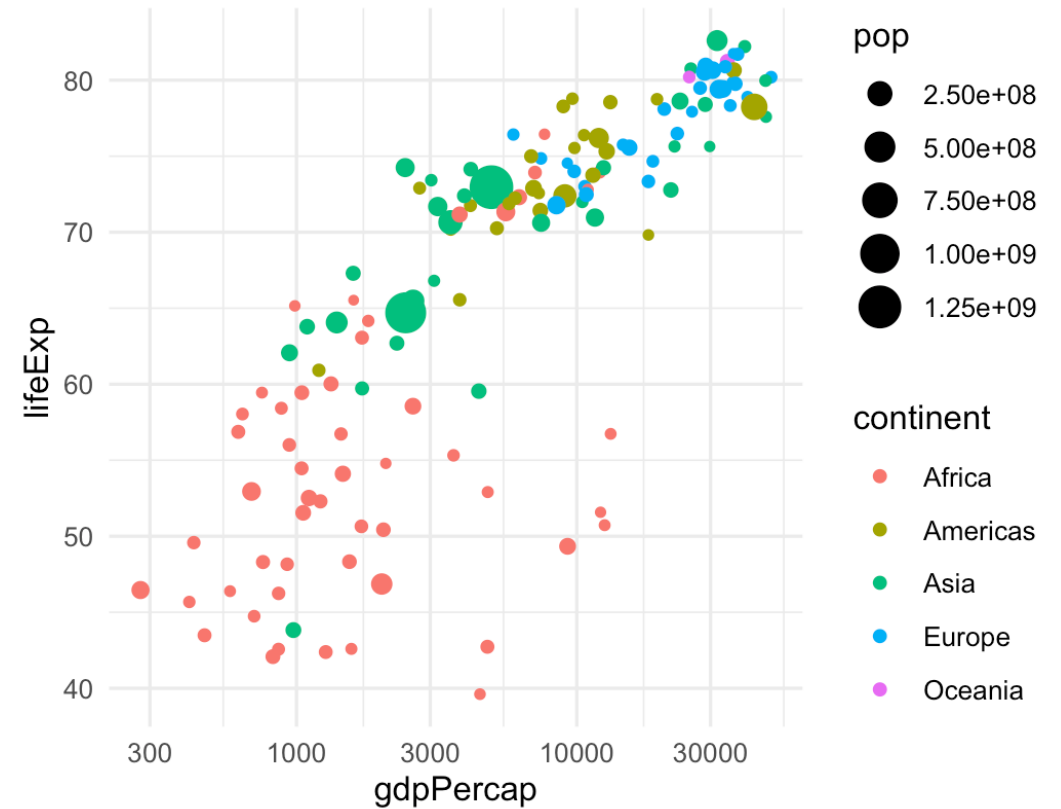
Example layer	What it does
<code>theme_grey()</code>	Default grey background
<code>theme_bw()</code>	Black and white
<code>theme_dark()</code>	Dark
<code>theme_minimal()</code>	Minimal

Theme

`theme_dark()`



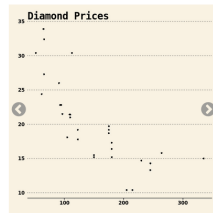
`theme_minimal()`



Theme

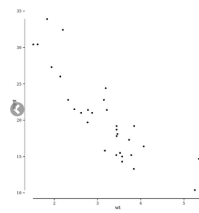
There are collections of pre-built themes online,
like the **{ggthemes}** package

ggthemes



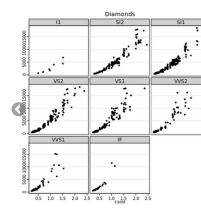
theme_wsj

Wall Street Journal theme



theme_tufte

Tufte Maximal Data, Minimal Ink Theme



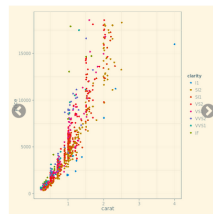
theme_stata

Themes based on Stata graph schemes



theme_solid

Theme with nothing other than a background color



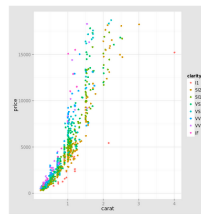
theme_solarized

ggplot color themes based on the Solarized palette



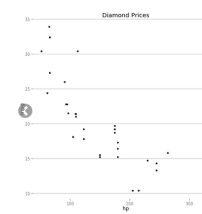
theme_map

Clean theme for maps



theme_igray

Inverse gray theme



theme_hc

Highcharts JS theme

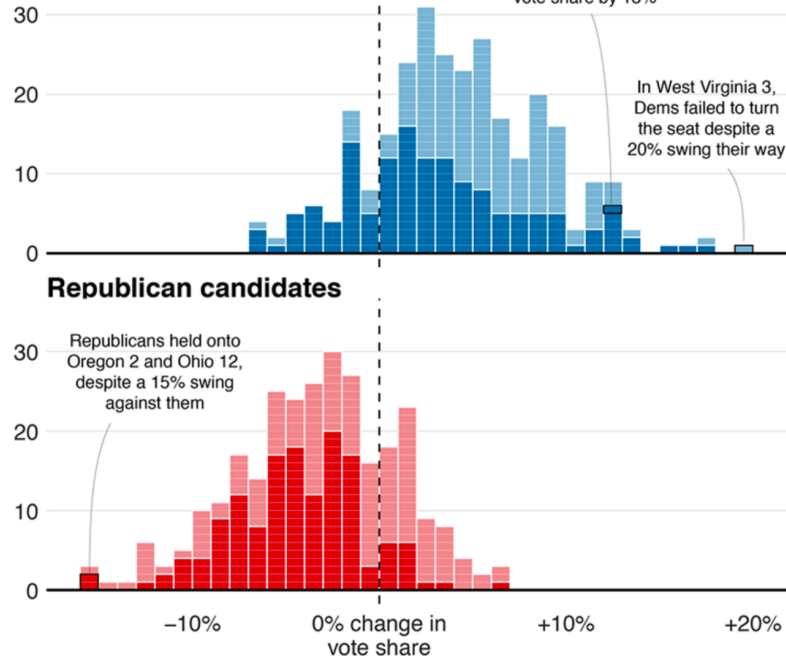
Theme

Organizations often make their own custom themes, like the BBC

Blue wave

■ Won seat ■ Didn't win

Democrat candidates

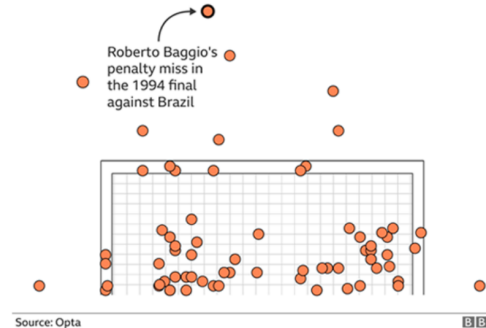


Source: AP, 19:01 ET

BBC

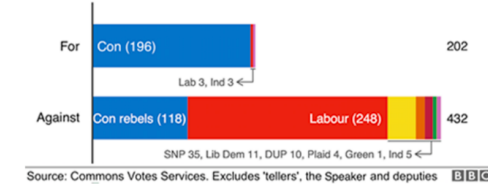
Where penalties are saved

World Cup shootout misses and saves, 1982-2014



Source: Opta

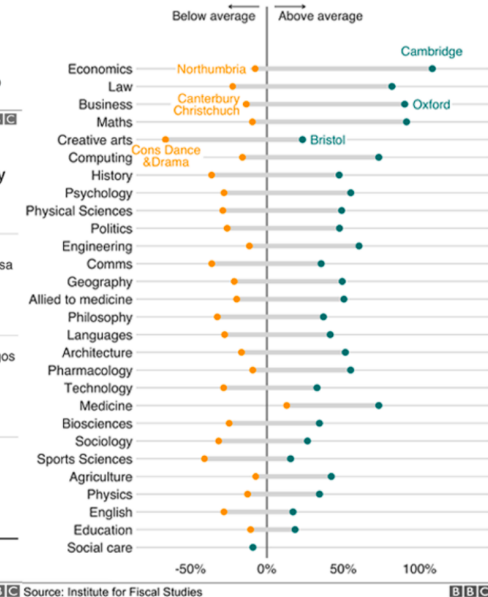
MPs rejected Theresa May's deal by 230 votes



Source: Commons Votes Services. Excludes 'tellers', the Speaker and deputies

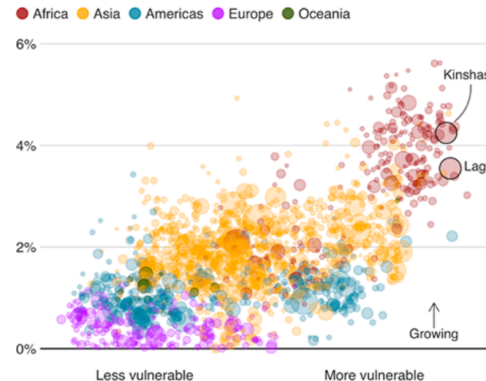
Earnings vary across unis even within subjects

Impact on men's earnings relative to the average degree



Fast-growing cities face worse climate risks

Population growth 2018-2035 over climate change vulnerability



Source: Verisk Maplecroft. Circle size represents current population.

Source: Institute for Fiscal Studies

BBC

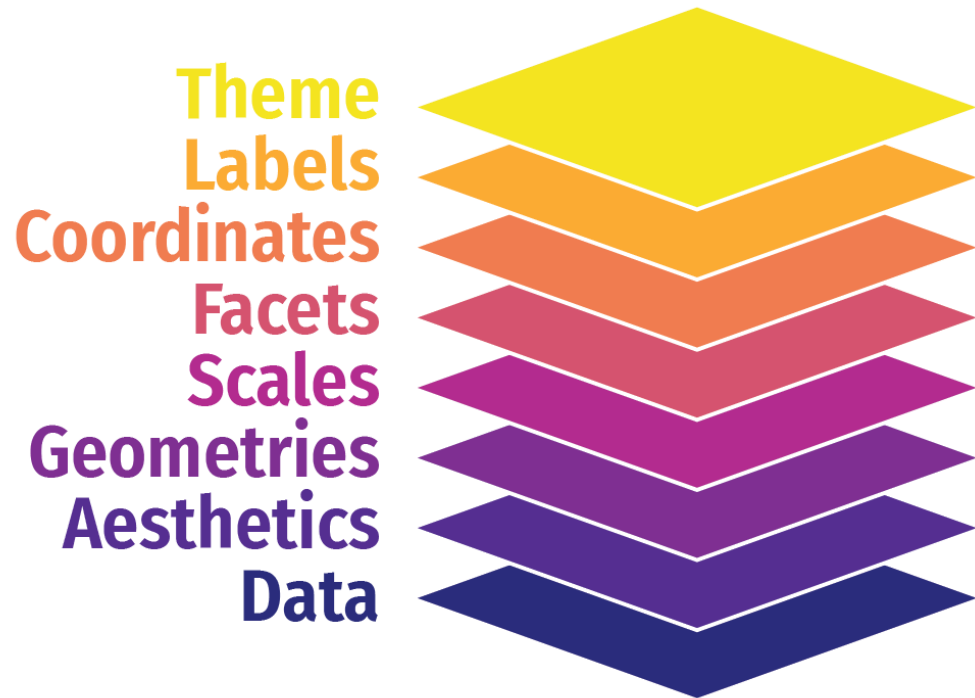
Theme options

Make theme adjustments with `theme()`

There are a billion options here!
We have a whole class session dedicated to this!

```
theme_bw() +  
theme(legend.position = "bottom",  
      plot.title = element_text(face = "bold"),  
      panel.grid = element_blank(),  
      axis.title.y = element_text(face = "italic"))
```

So many possibilities!



These were just a few examples of layers!

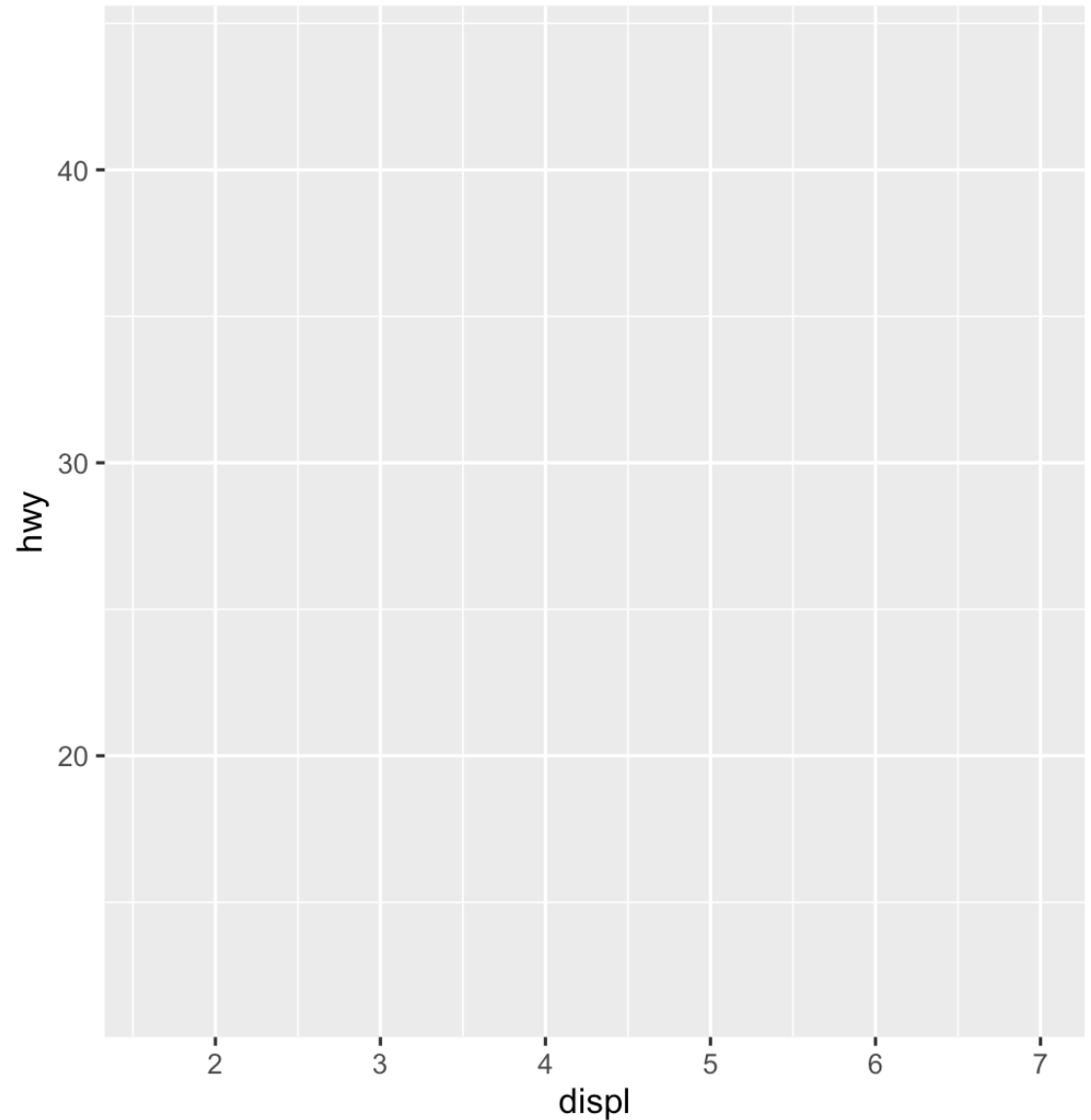
See the `{ggplot2}` documentation for complete examples of everything you can do

Putting it all together

**We can build a plot sequentially
to see how each grammatical layer
changes the appearance**

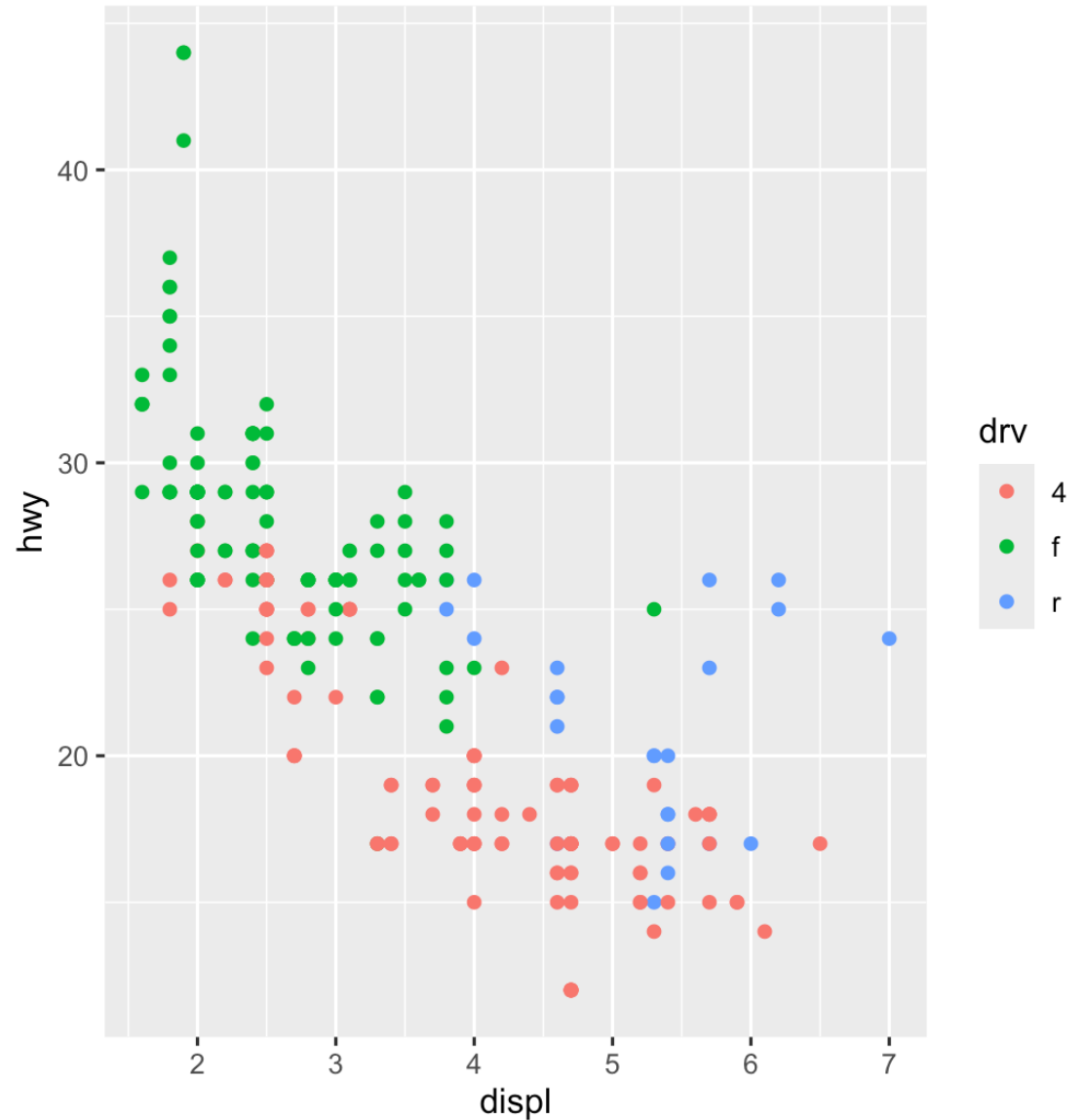
Start with data and aesthetics

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



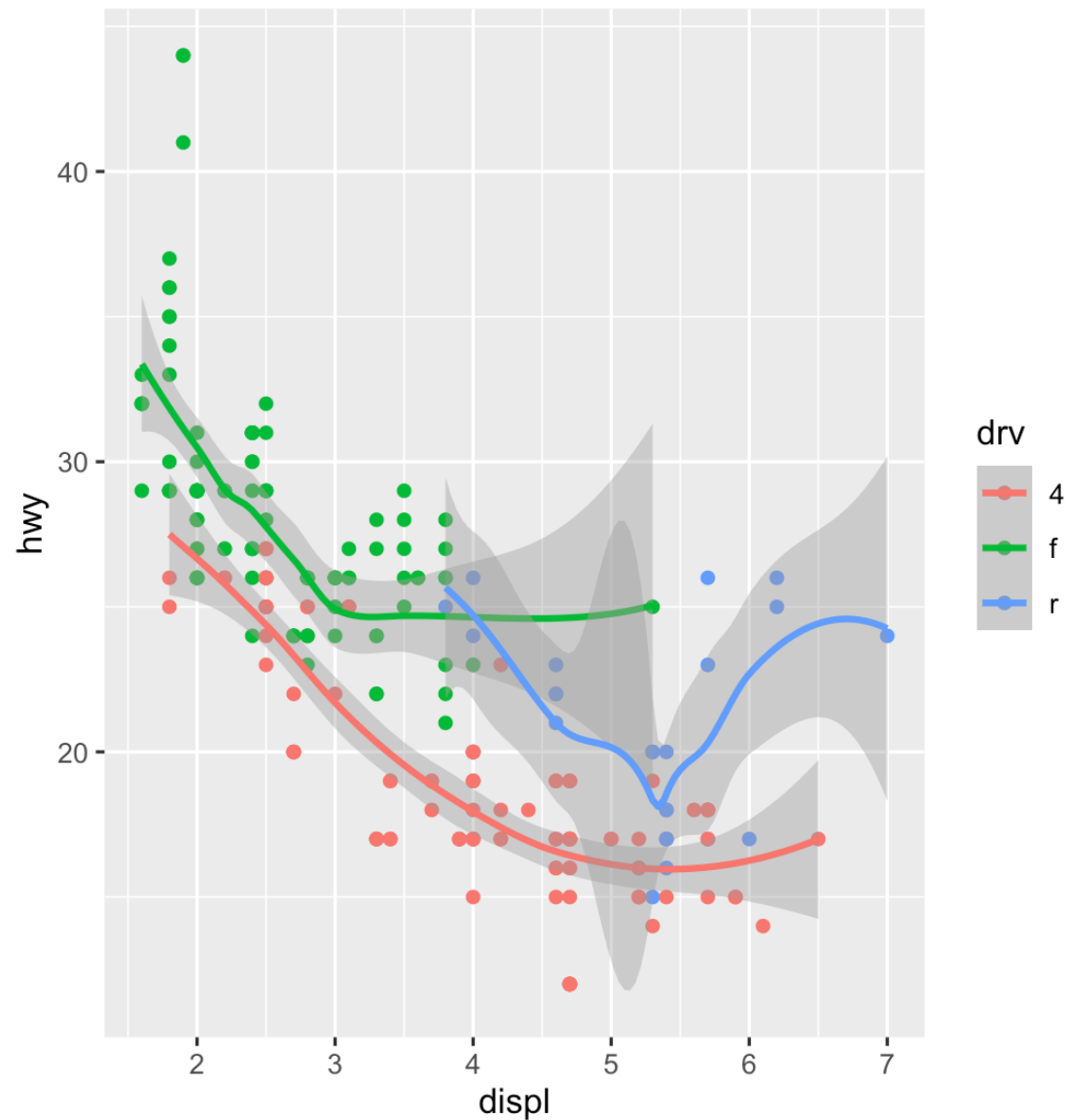
Add a point geom

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



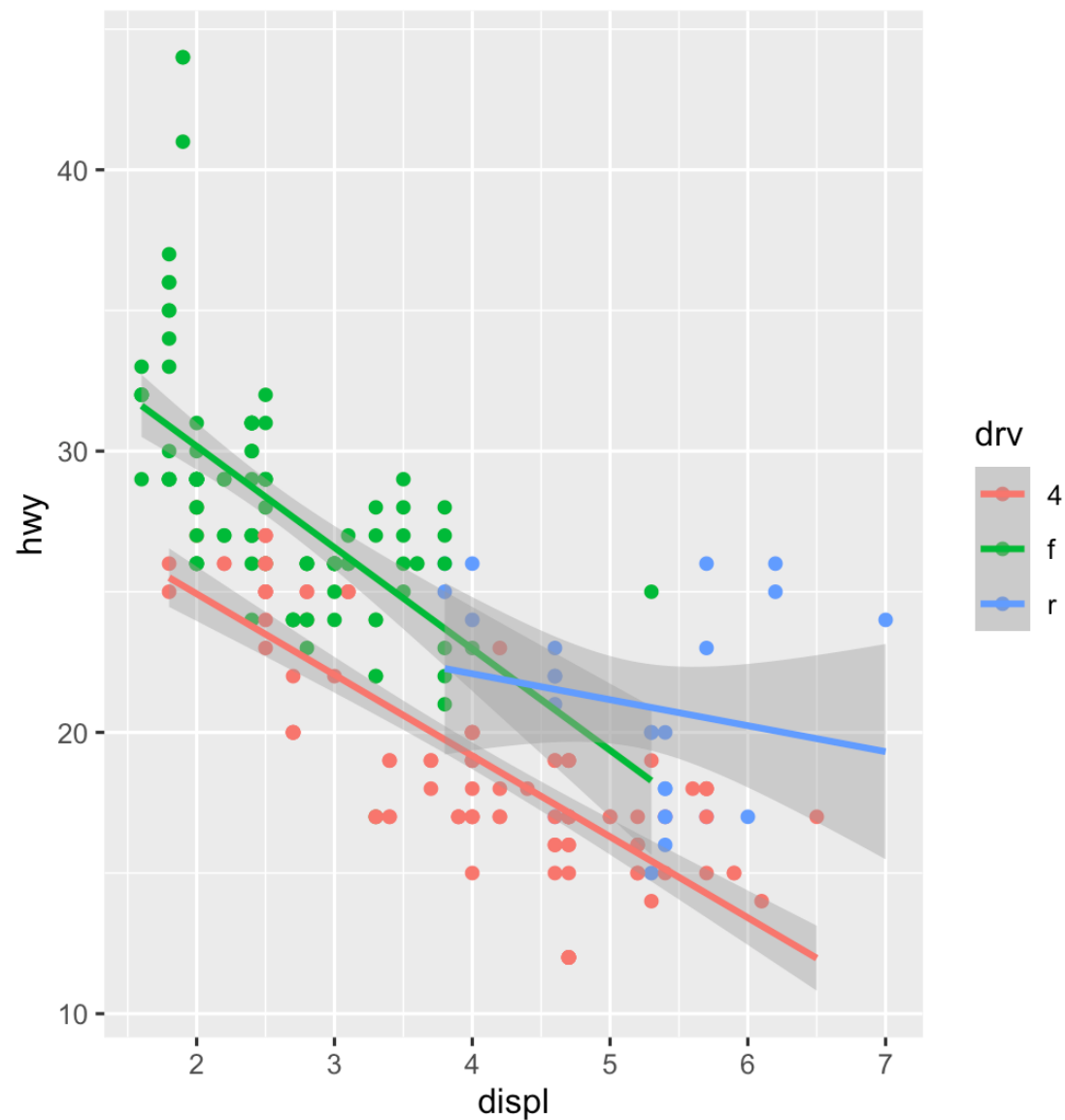
Add a smooth geom

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



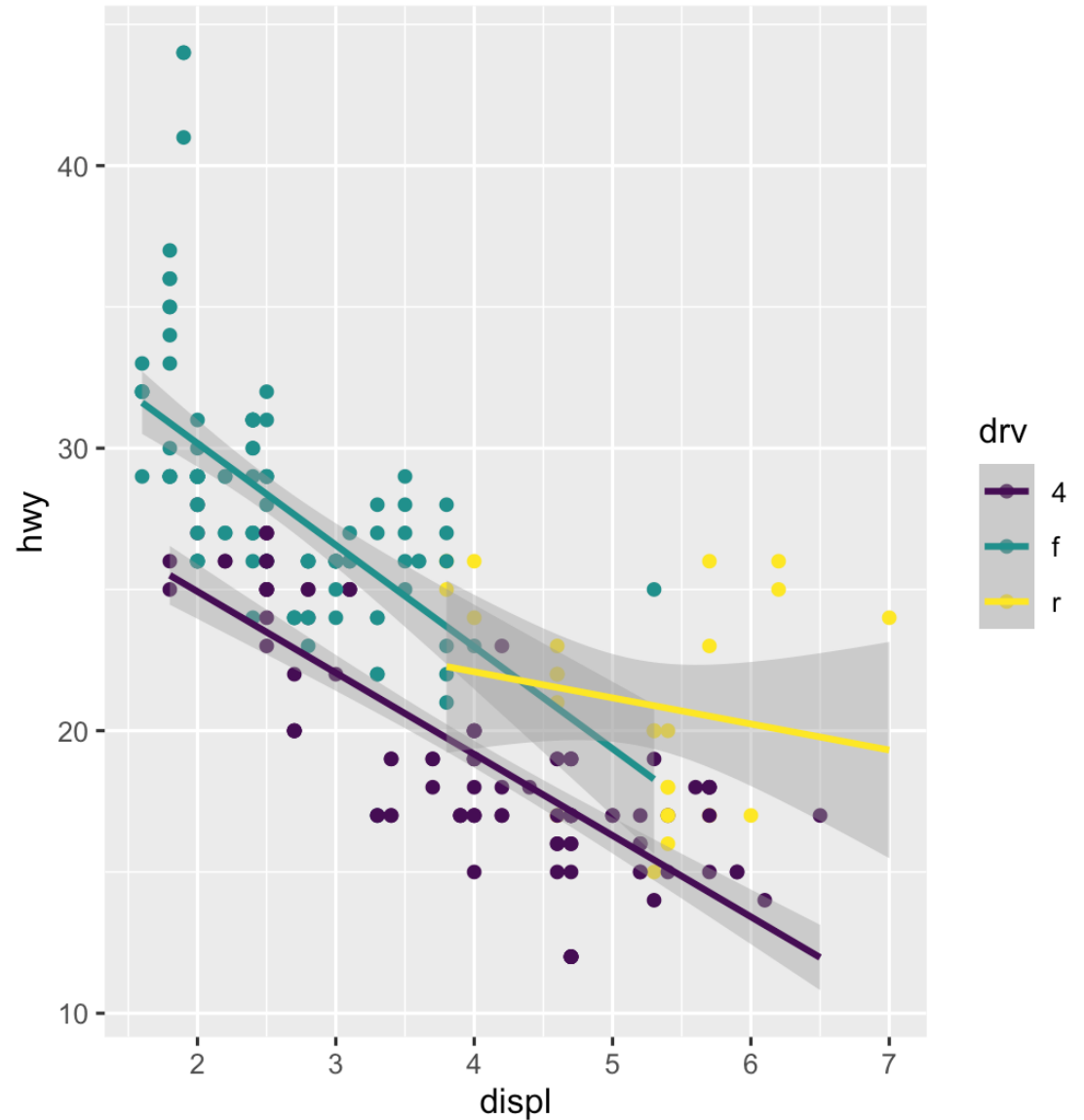
Make it straight

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



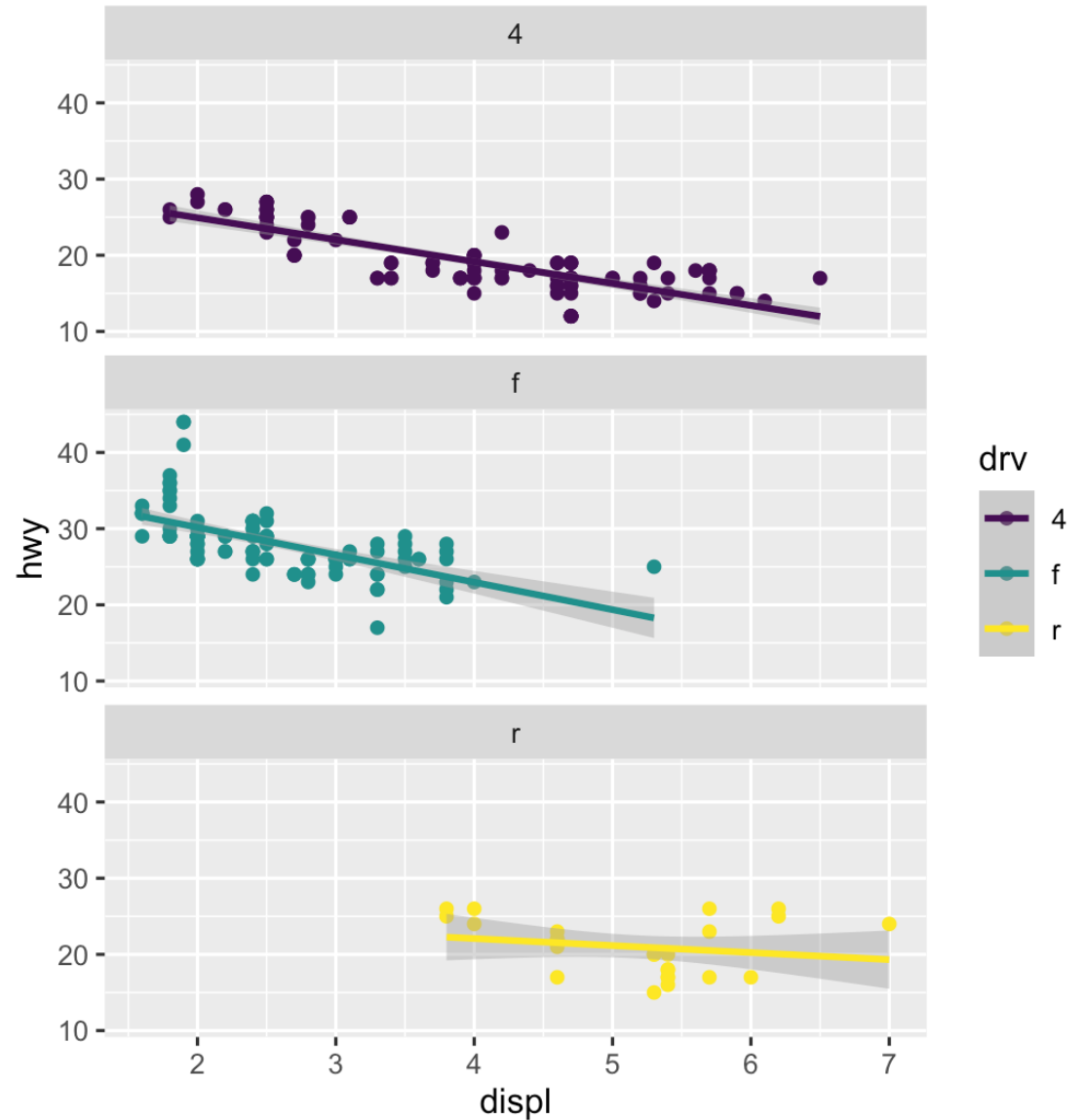
Use a viridis color scale

```
ggplot(data = mpg,  
       mapping = aes(x = displ,  
                     y = hwy,  
                     color = drv)) +  
  geom_point() +  
  geom_smooth(method = "lm") +  
  scale_color_viridis_d()
```



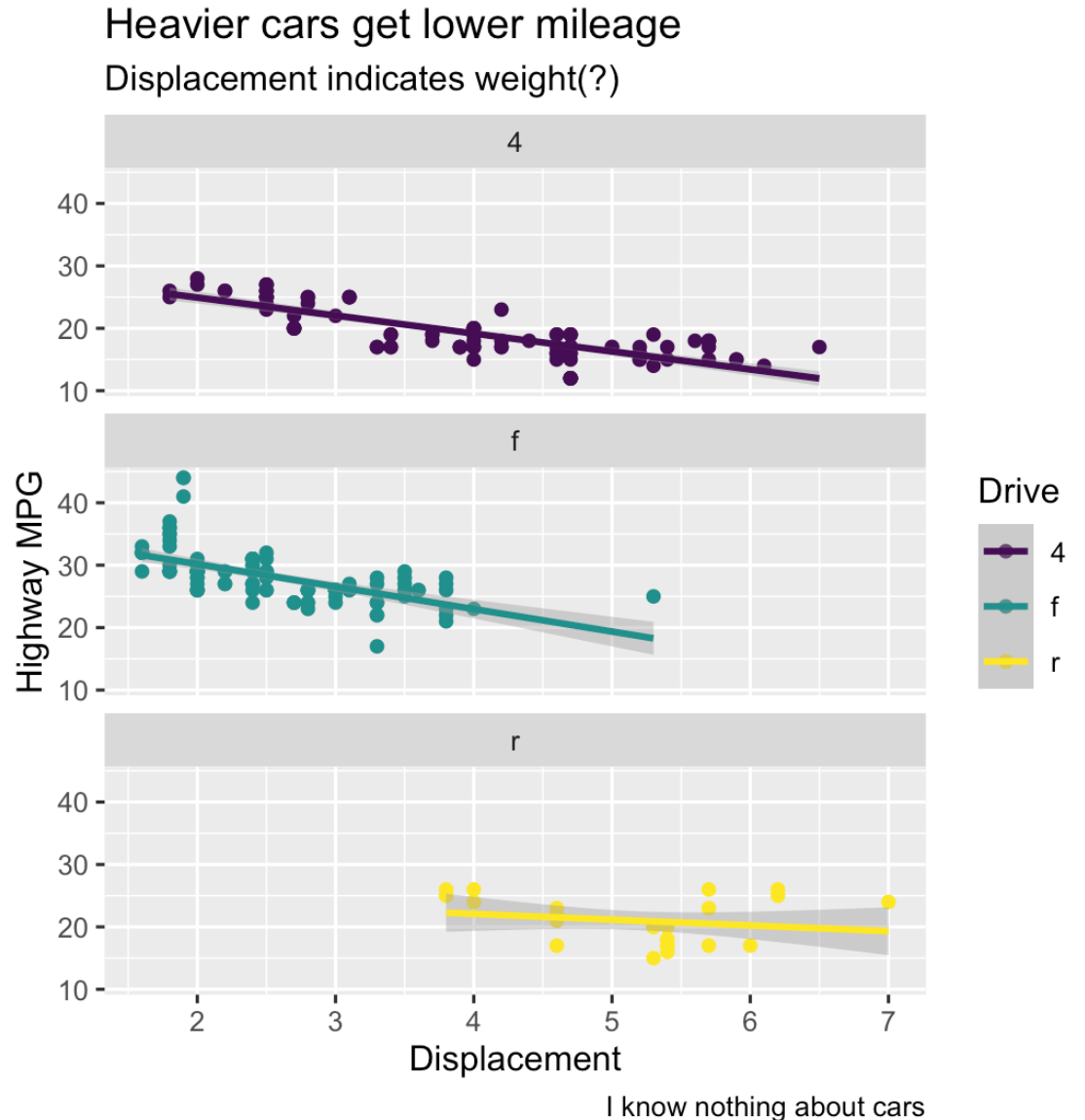
Facet by drive

```
ggplot(data = mpg,  
       mapping = aes(x = displ,  
                     y = hwy,  
                     color = drv)) +  
  geom_point() +  
  geom_smooth(method = "lm") +  
  scale_color_viridis_d() +  
  facet_wrap(vars(drv), ncol = 1)
```



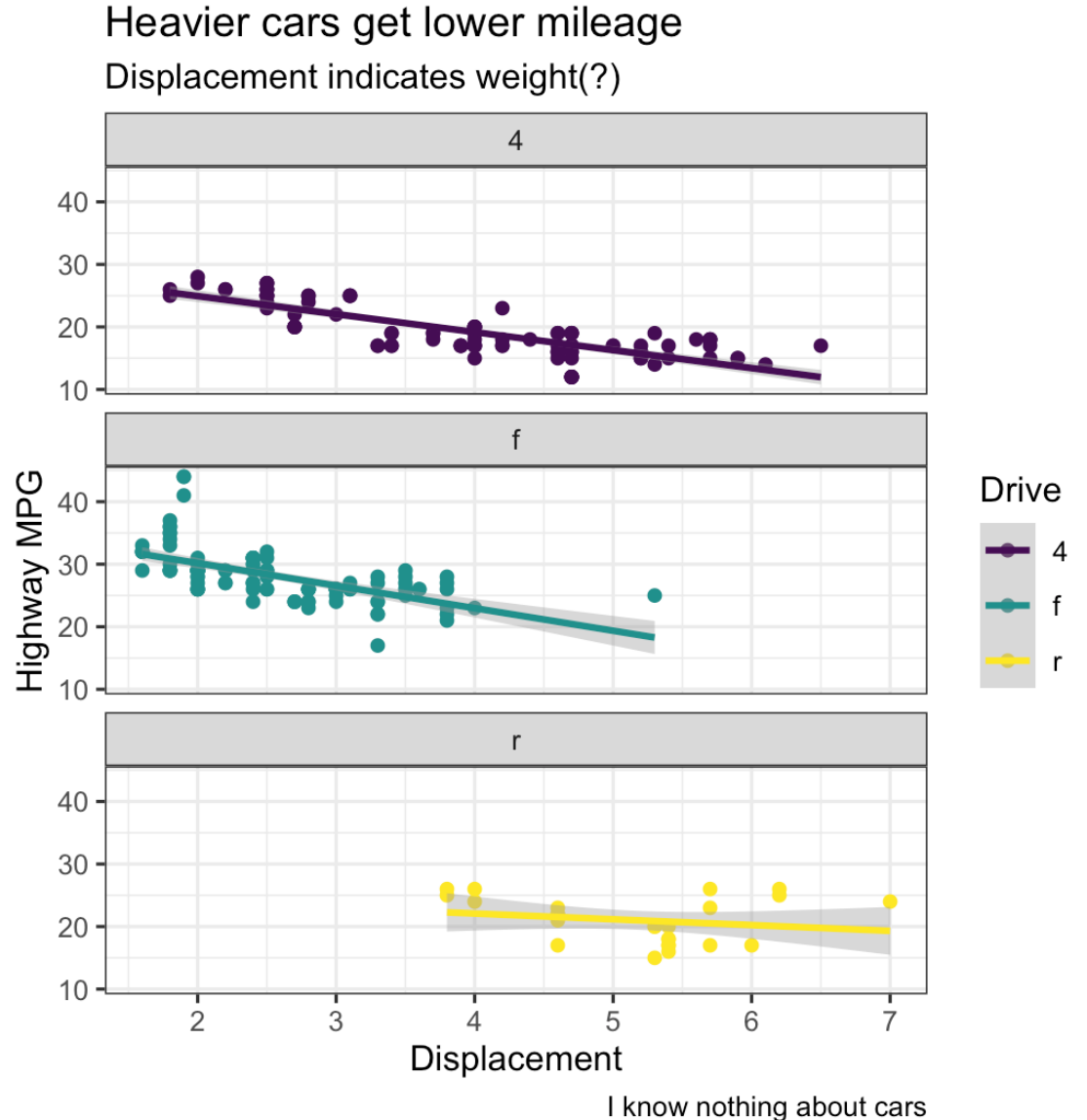
Add labels

```
ggplot(data = mpg,
       mapping = aes(x = displ,
                     y = hwy,
                     color = drv)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(drv), ncol = 1) +
  labs(x = "Displacement", y = "Highway MPG",
       color = "Drive",
       title = "Heavier cars get lower mileage",
       subtitle = "Displacement indicates weight",
       caption = "I know nothing about cars")
```



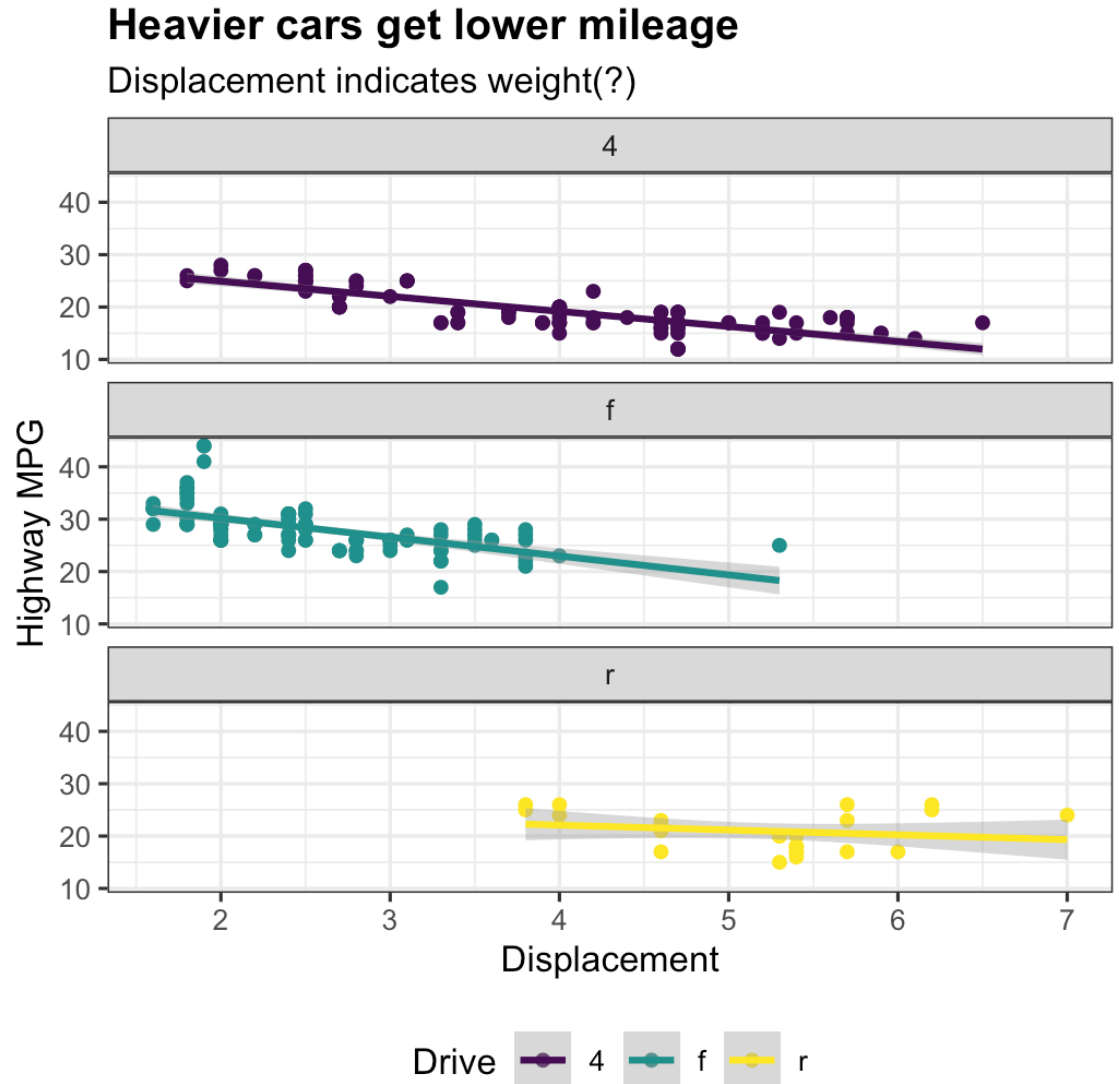
Add a theme

```
ggplot(data = mpg,
       mapping = aes(x = displ,
                     y = hwy,
                     color = drv)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(drv), ncol = 1) +
  labs(x = "Displacement", y = "Highway MPG",
       color = "Drive",
       title = "Heavier cars get lower mileage",
       subtitle = "Displacement indicates weight",
       caption = "I know nothing about cars") +
  theme_bw()
```



Modify the theme

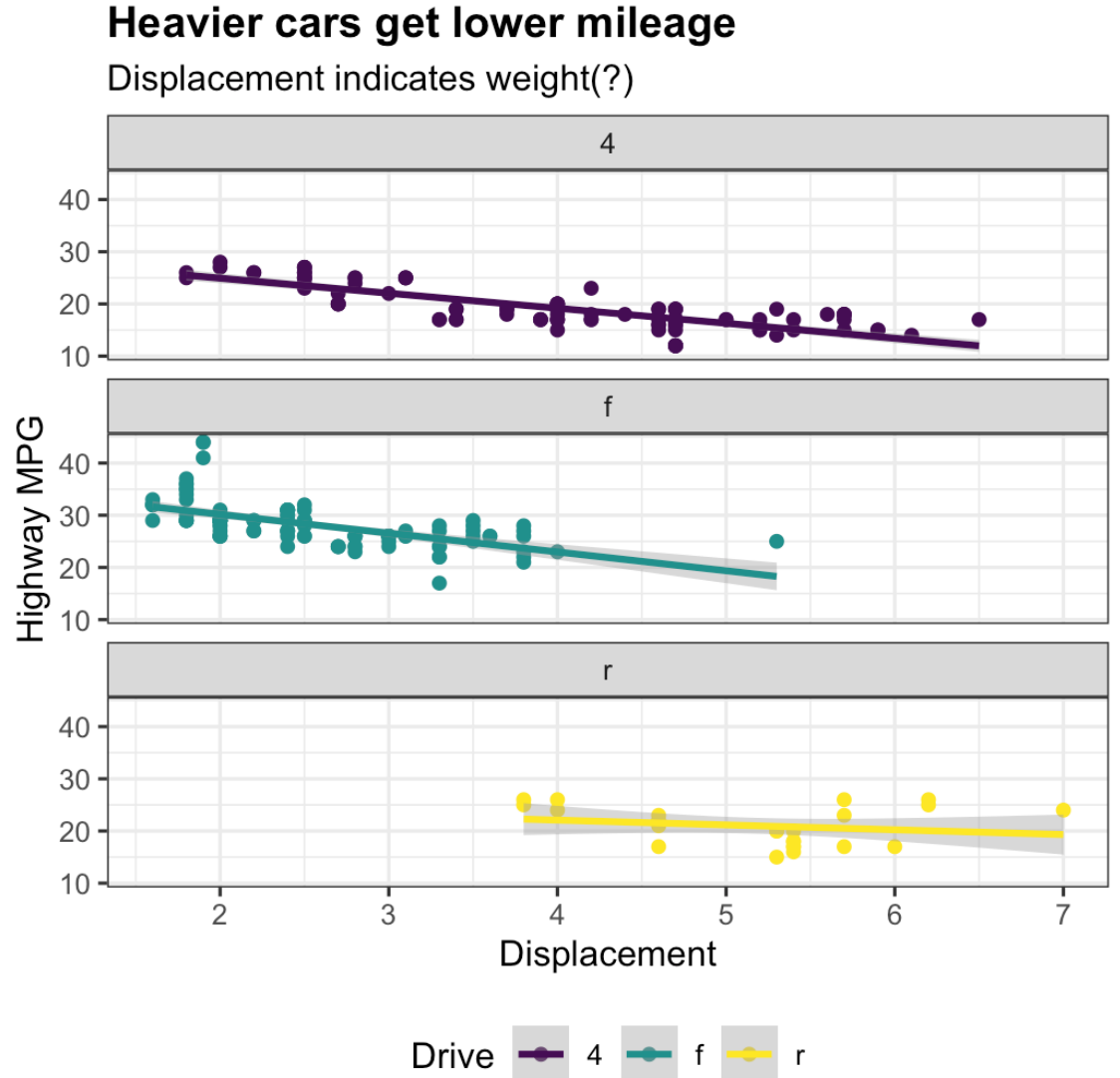
```
ggplot(data = mpg,
       mapping = aes(x = displ,
                     y = hwy,
                     color = drv)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(drv), ncol = 1) +
  labs(x = "Displacement", y = "Highway MPG",
       color = "Drive",
       title = "Heavier cars get lower mileage",
       subtitle = "Displacement indicates weight",
       caption = "I know nothing about cars") +
  theme_bw() +
  theme(legend.position = "bottom",
        plot.title = element_text(face = "bold"))
```



I know nothing about cars

Finished!

```
ggplot(data = mpg,
       mapping = aes(x = displ,
                     y = hwy,
                     color = drv)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(drv), ncol = 1) +
  labs(x = "Displacement", y = "Highway MPG",
       color = "Drive",
       title = "Heavier cars get lower mileage",
       subtitle = "Displacement indicates weight",
       caption = "I know nothing about cars") +
  theme_bw() +
  theme(legend.position = "bottom",
        plot.title = element_text(face = "bold"))
```

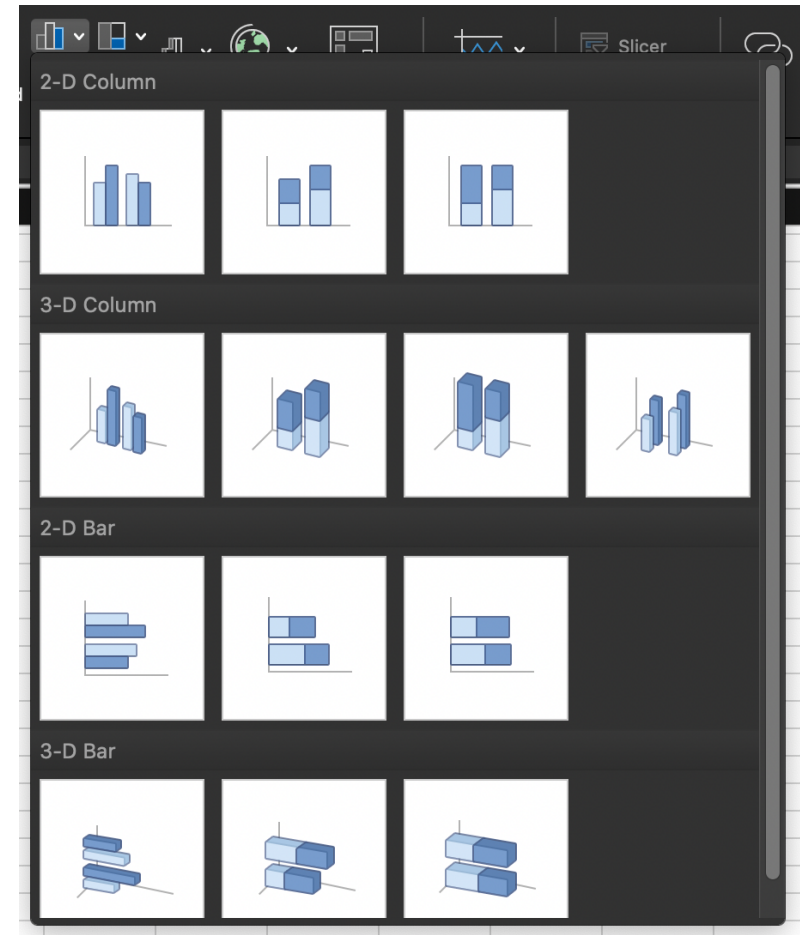


I know nothing about cars

A true grammar

With the grammar of graphics,
we don't talk about specific
chart *types*

Hunt through Excel menus for a
stacked bar chart and manually
reshape your data to work with it

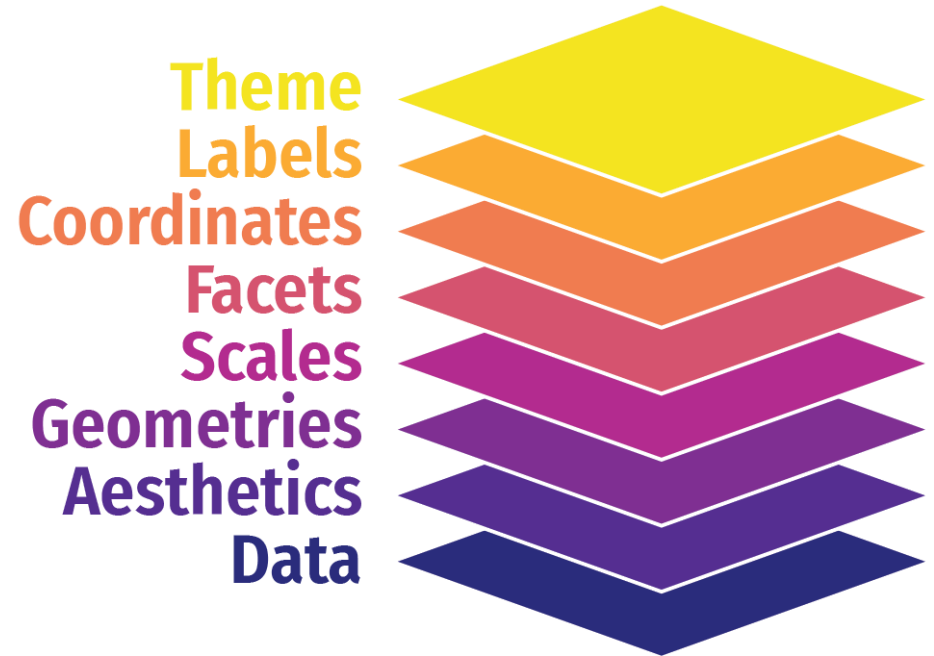


A true grammar

With the grammar of graphics,
we *do* talk about specific
chart *elements*

Map a column to the x-axis, fill by a
different variable, and `geom_col()` to
get stacked bars

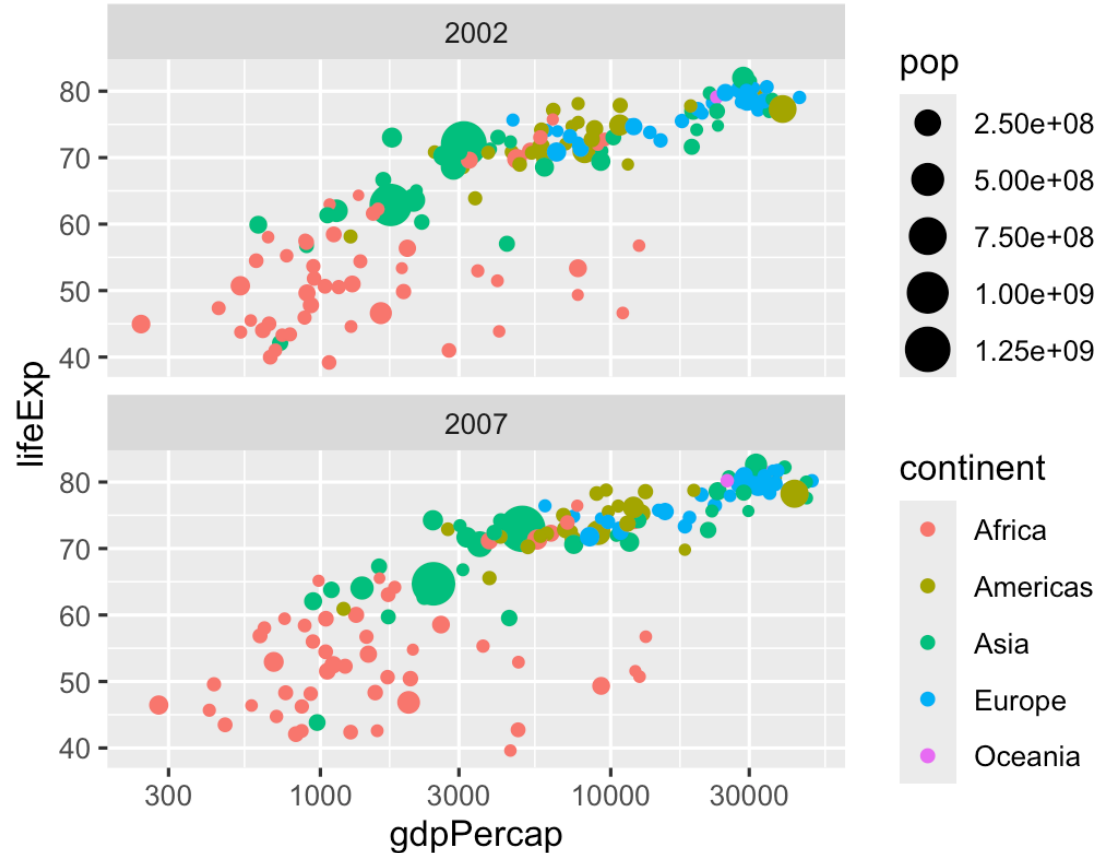
Geoms can be interchangeable
(e.g. switch `geom_violin()` to
`geom_boxplot()`)



Describing graphs with the grammar

Map wealth to the x-axis, health to the y-axis, add points, color by continent, size by population, scale the y-axis with a log, and facet by year

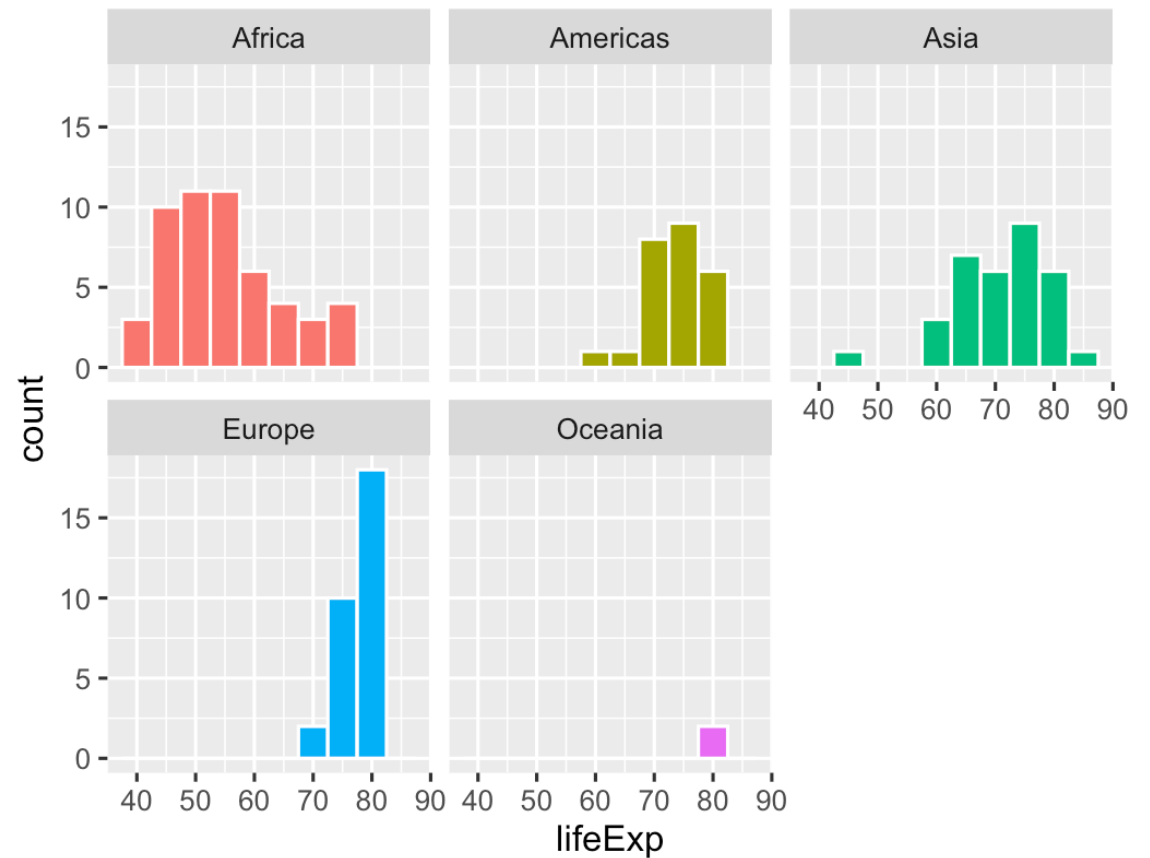
```
ggplot(data = filter(gapminder, year %in% c(2002, 2007))  
  mapping = aes(x = gdpPercap,  
    y = lifeExp,  
    color = continent,  
    size = pop)) +  
  geom_point() +  
  scale_x_log10() +  
  facet_wrap(vars(year), ncol = 1)
```



Describing graphs with the grammar

Map health to the x-axis, add a histogram with bins for every 5 years, fill and facet by continent

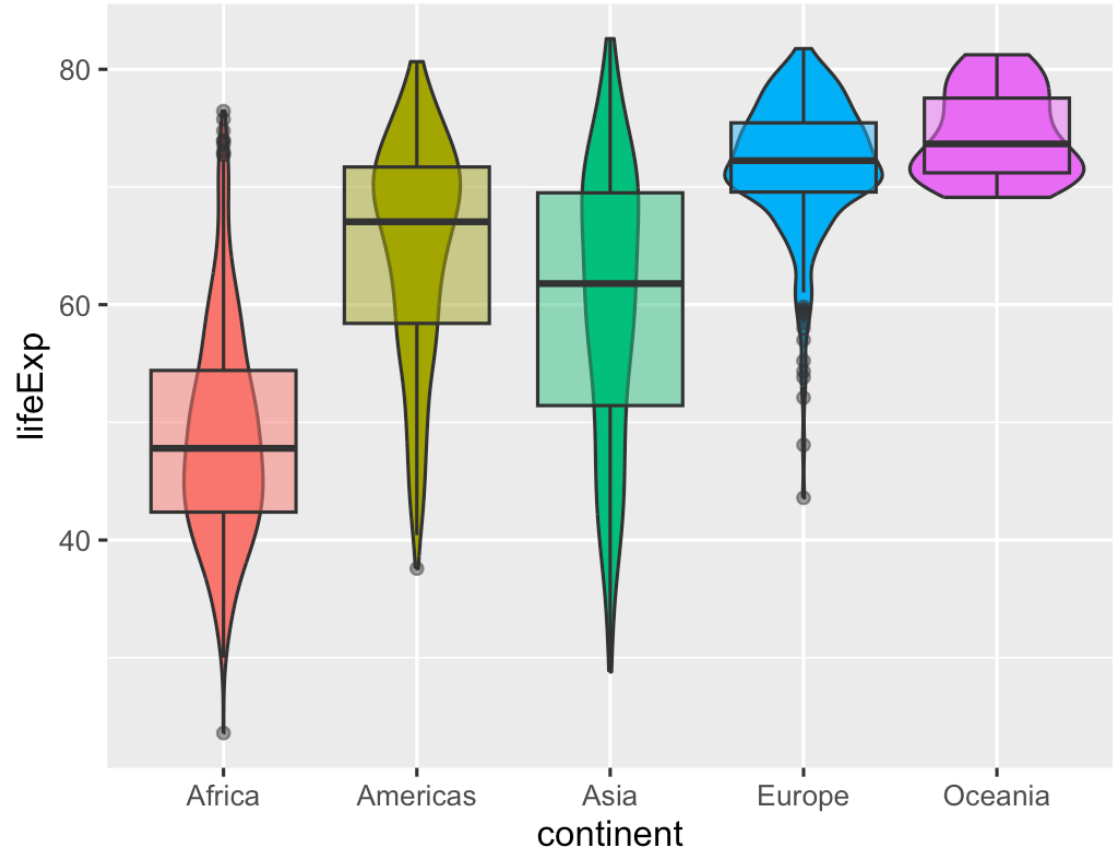
```
ggplot(data = gapminder_2007,  
       mapping = aes(x = lifeExp,  
                     fill = continent)) +  
  geom_histogram(binwidth = 5,  
                color = "white") +  
  guides(fill = "none") + # Turn off legend  
  facet_wrap(vars(continent))
```



Describing graphs with the grammar

Map continent to the x-axis, health to the y-axis, add violin plots and semi-transparent boxplots, fill by continent

```
ggplot(data = gapminder,  
       mapping = aes(x = continent,  
                     y = lifeExp,  
                     fill = continent)) +  
  geom_violin() +  
  geom_boxplot(alpha = 0.5) +  
  guides(fill = "none") # Turn off legend
```

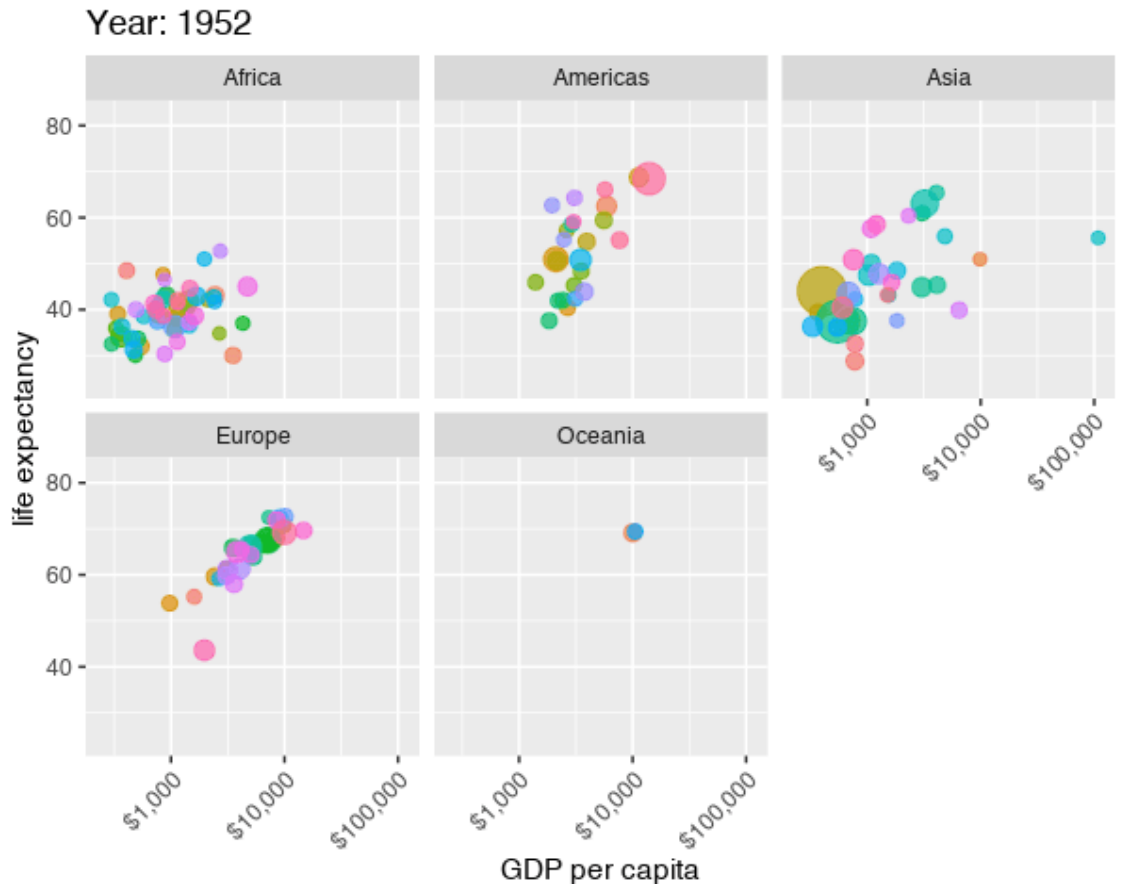


Aesthetics in extra dimensions

Time

Use {gganimate} to map variables to a time aesthetic

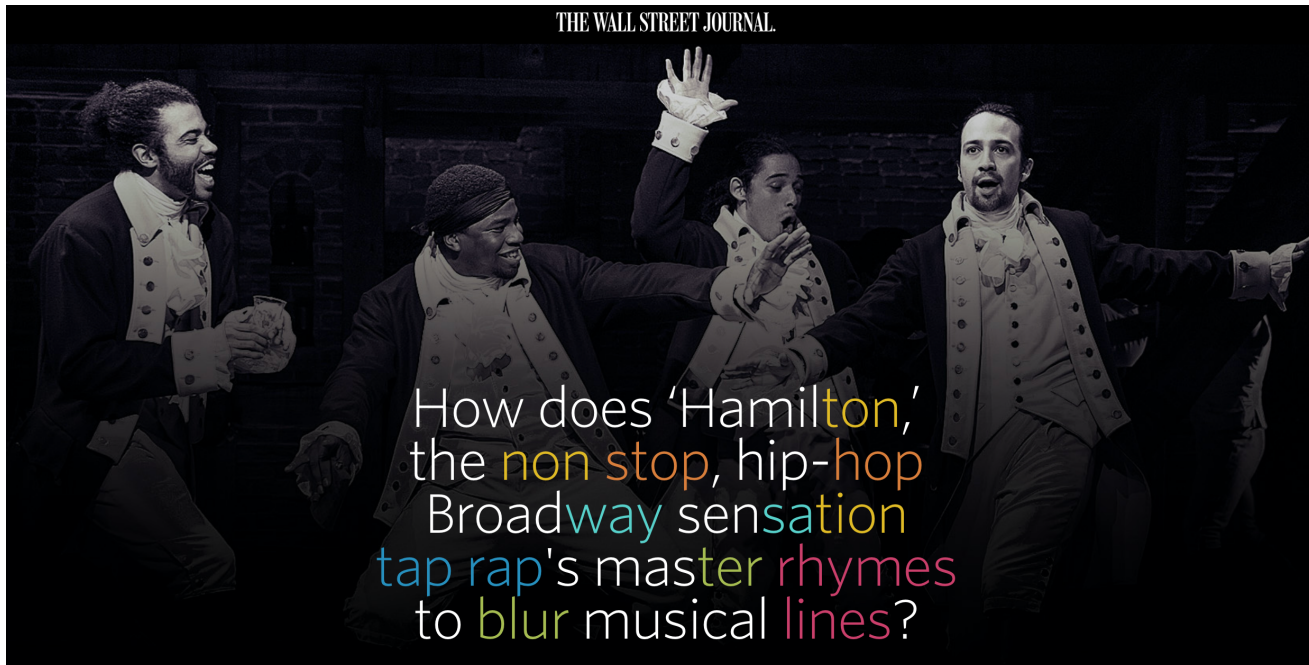
```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, size = pop, color = continent)) +  
  geom_point(alpha = 0.7) +  
  scale_size(range = c(2, 12)) +  
  scale_x_log10(labels = scales::label_dollar) +  
  guides(size = "none", color = "none") +  
  facet_wrap(~continent) +  
  # Special gganimate stuff  
  labs(title = 'Year: {frame_time}', x = 'GDP per capita') +  
  transition_time(year) +  
  ease_aes('linear')
```



Sound

Visualize internal rhyming schemes in music

<http://graphics.wsj.com/hamilton/>



IH-N
AE
IH
W-AA
AH
IT

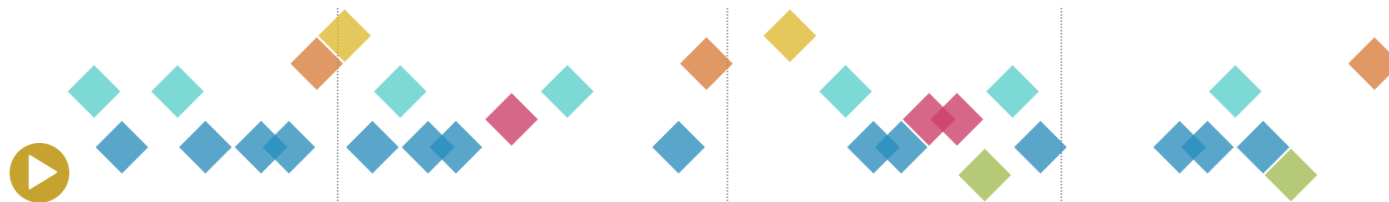


Daveed Diggs

I'm **in** the **cabinet** I **am** **complicit in**
Watching him **grabbing** at power **and** **kissing it**
If **Washington** isn't gon' listen
To **disciplined dissidents this is** the **difference**
This kid **is** out

"Washington On Your Side" on "Hamilton (Original Broadway Cast Recording)"

AE
EY
AY
UW
AH
IY



Kendrick Lamar

Trapped inside your **desire** to **fire** bullets **that stray**
Track attire just tell **you** I'm **tired** and ran **away**
I should **ask** a **choir** "**What do you require**
to sing a song **that acquire me** to have **faith?**"

"good kid" on "good kid, m.A.A.d city"

Animation, time, and sound

The Weather Channel Uses Animation to Show Dangers of Storm Surge



Tidy data

Data shapes

For `ggplot()` to work,
your data needs to be in a **tidy** format

This doesn't mean that it's clean—
it refers to the *structure* of the data

All the packages in the {tidyverse} work best with
tidy data; that's why it's called that!

Tidy data

Each variable has its own column

Each observation has its own row

Each value has its own cell

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	866	20593360
Brazil	1999	3737	172006362
Brazil	2000	8488	17404898
China	1999	21258	1272015272
China	2000	21476	128042583

variables

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	866	20593360
Brazil	1999	3737	172006362
Brazil	2000	8488	17404898
China	1999	21258	1272015272
China	2000	21476	128042583

observations

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	866	20593360
Brazil	1999	3737	172006362
Brazil	2000	8488	17404898
China	1999	21258	1272015272
China	2000	21476	128042583

values

From chapter 12 of *R for Data Science*

Untidy data example

Real world data is often untidy, like this:

	A	B	C	D
1	Number of incidents			
2				
3	Office	2015	2016	2017
4	Utah County	134	145	167
5	Salt Lake County	302	334	331
6	Davis County	254	288	299
7	Juab County	78	82	87
8				
9	bold = needs verification			
10	yellow = compiled from different source			
11				

Tidy data example

Here's the tidy version of that same data:

	A	B	C	D	E
1	Office	Year	Incidents	Needs Verification	Different Source
2	Utah County	2015	134	FALSE	FALSE
3	Salt Lake County	2015	302	TRUE	FALSE
4	Davis County	2015	254	FALSE	FALSE
5	Juab County	2015	78	FALSE	FALSE
6	Utah County	2016	145	FALSE	TRUE
7	Salt Lake County	2016	334	FALSE	FALSE
8	Davis County	2016	288	FALSE	FALSE
9	Juab County	2016	82	TRUE	TRUE
10	Utah County	2017	167	TRUE	FALSE
11	Salt Lake County	2017	331	FALSE	FALSE
12	Davis County	2017	299	FALSE	TRUE
13	Juab County	2017	87	FALSE	FALSE

This is plottable!

Wide vs. long

Tidy data is also called "long" data

wide

id	x	y	z
1	a	c	e
2	b	d	f

long

id	key	val
1	x	a
2	x	b
1	y	c
2	y	d
1	z	e
2	z	f

Moving from wide to long

Nowadays, `gather()` is called `pivot_longer()` and `spread()` is called `pivot_wider()`

wide

id	wide		
	x	y	z
1	a	c	e
2	b	d	f