Intro to ggplot2

Getting started

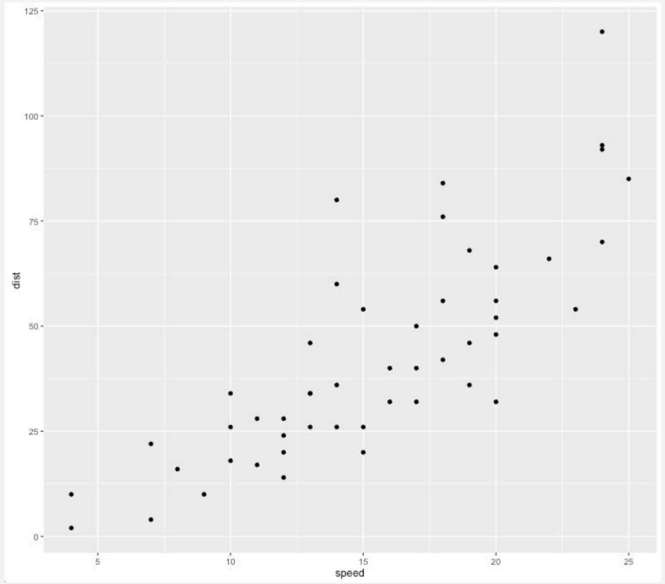
```
install.packages("ggplot2")
```

And on any script you want to use it …

```
require(ggplot2)
```

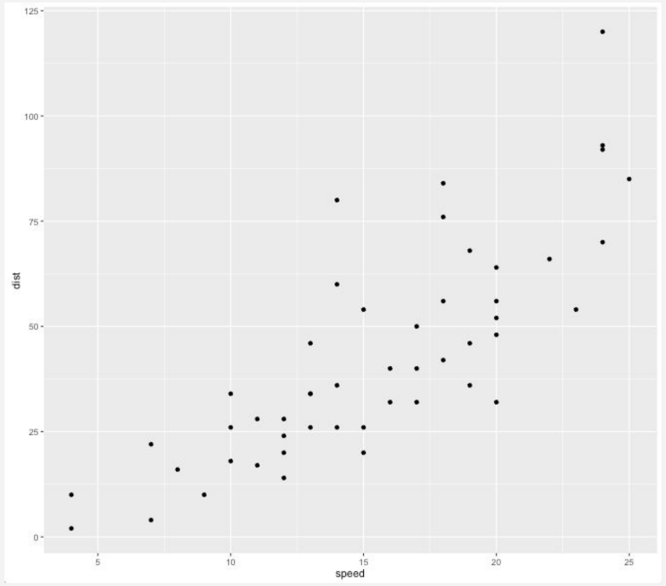
Basic syntax: ggplot, data, geom, aes

```
chart <- ggplot(data=cars, aes(x=speed, y=dist))+
    geom_point()

print(chart)
```



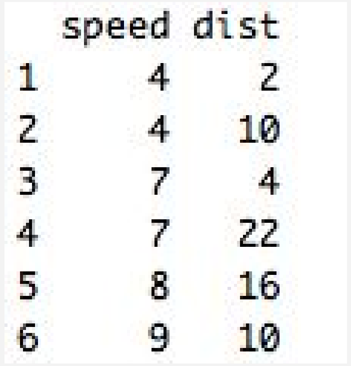
Basic syntax: ggplot, data, geom, aes

```
chart <- ggplot(data=cars, aes(x=speed, y=dist))+
    geom_point()

print(chart)
```

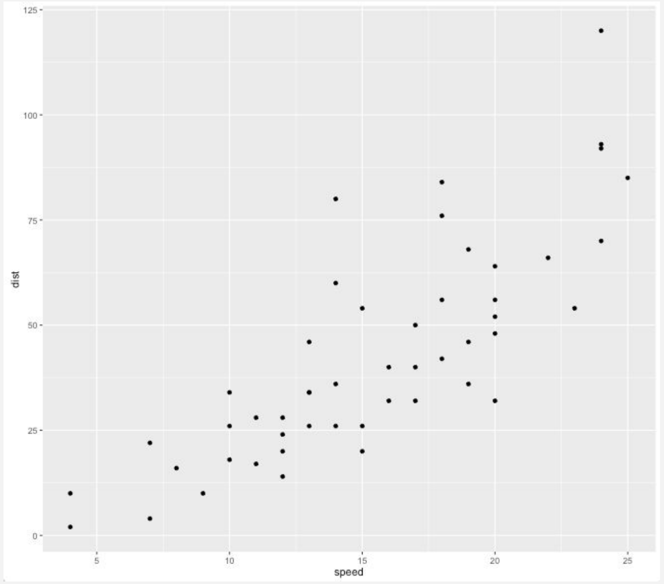


Basic syntax: ggplot, data, geom, aes

```
chart <- ggplot(data=cars, aes(x=speed, y=dist))+
    geom_point()

print(chart)
```

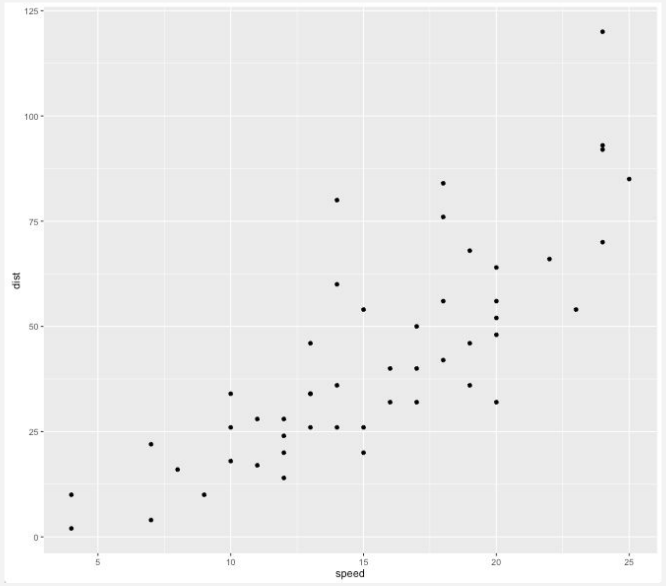


```
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```



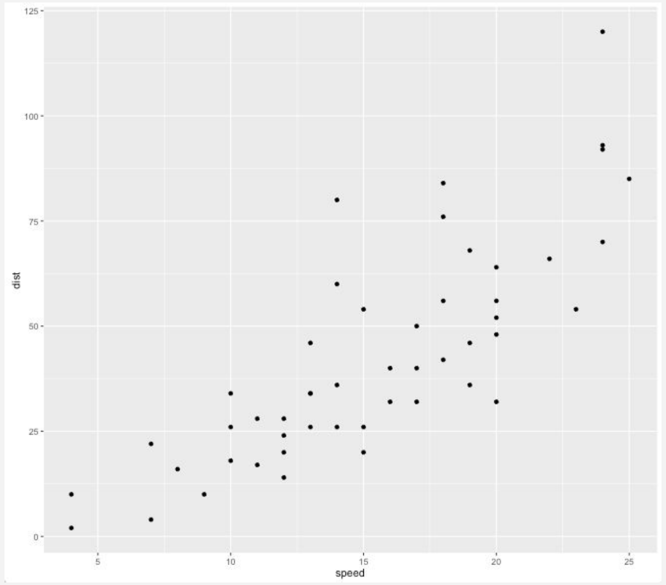
Basic syntax: ggplot, data, geom, aes

```
chart <- ggplot(data=cars, aes(x=speed, y=dist))+
    geom_point()

print(chart)
```



Basic syntax: ggplot, data, geom, aes

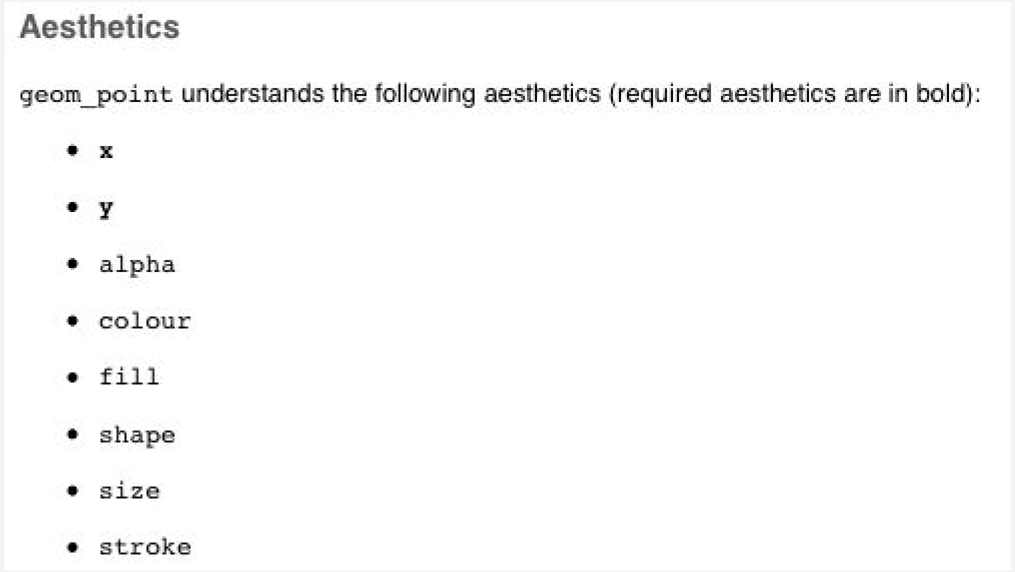
```
chart <- ggplot(data=cars, aes(x=speed, y=dist))+
   geom_point()

print(chart)
```



How do I know what aes to use?

Different geoms require (and accept) different data aesthetics

```
?geom_point
```



Aesthetics

`geom_point` understands the following aesthetics (required aesthetics are in bold):

- **`x`**
- **`y`**
- `alpha`
- `colour`
- `fill`
- `shape`
- `size`
- `stroke`

Geoms will look "up the ladder" for aes

Snippets below will produce the same charts

```
ggplot(data=cars, aes(x=speed, y=dist))+
    geom_point()
```

```
ggplot(data=cars)+
    geom_point(aes(x=speed, y=dist))
```

(They'll do the same for data)

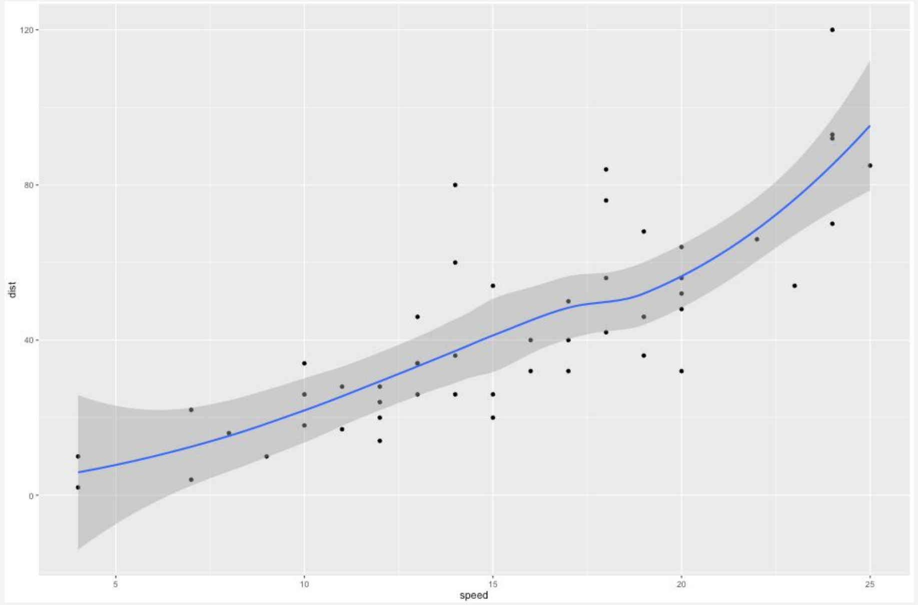
Two snippets below will produce the same chart

```
ggplot(data=cars, aes(x=speed, y=dist))+
   geom_point()


ggplot()+
   geom_point(data=cars, aes(x=speed, y=dist))
```

So why do we put all that stuff in ggplot()?

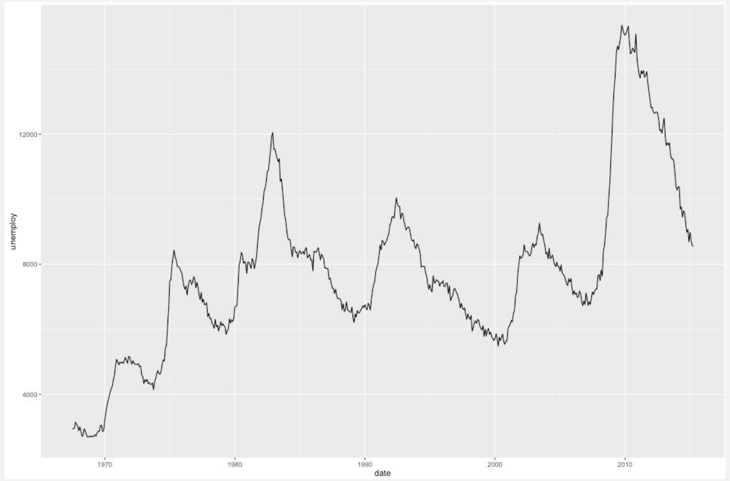
Charts can have multiple geoms, referencing the same aes!

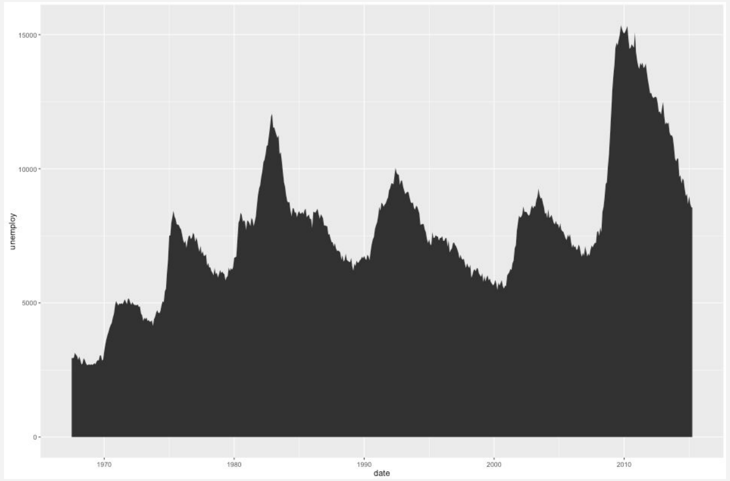
```
chart <- ggplot(data=cars, aes(x=speed, y=dist))+
    geom_point()+
    geom_smooth()

print(chart)
```



Let's make some basic charts!

1. Using the built-in `economics` dataframe and `geom_line()`, make a line chart of how unemployment (`unemply`) has changed over time (`date`)
2. Try the same thing as an area chart using `geom_area()`

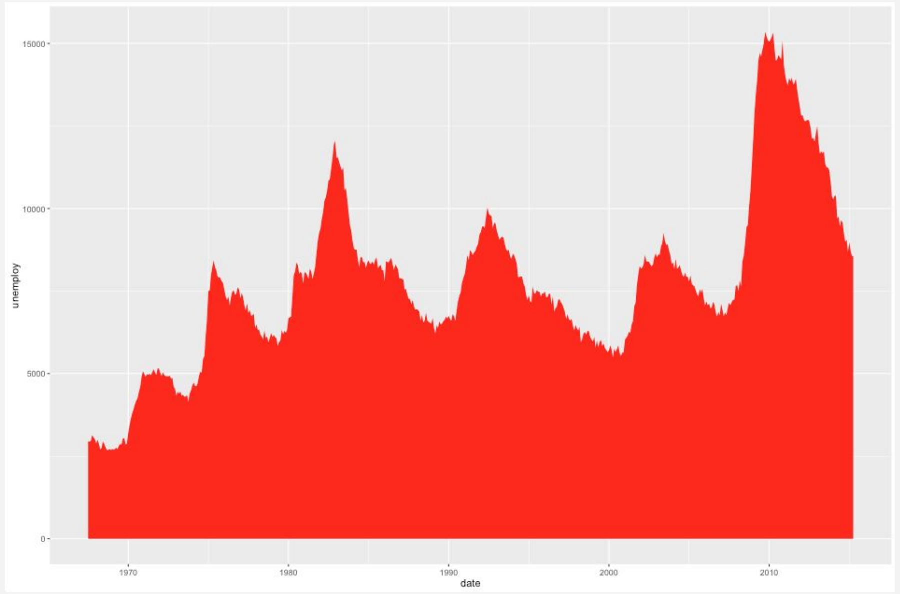




Prettier charts: Styling

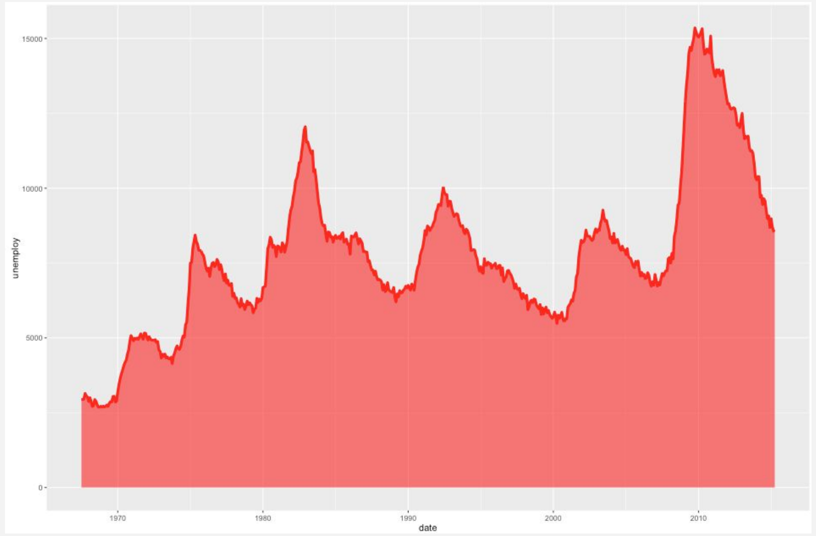
```r
chart <- ggplot(economics, aes(x=date, y=unemploy)) +
    geom_area(fill="red")
```

(Be careful of "fill" vs. "color")



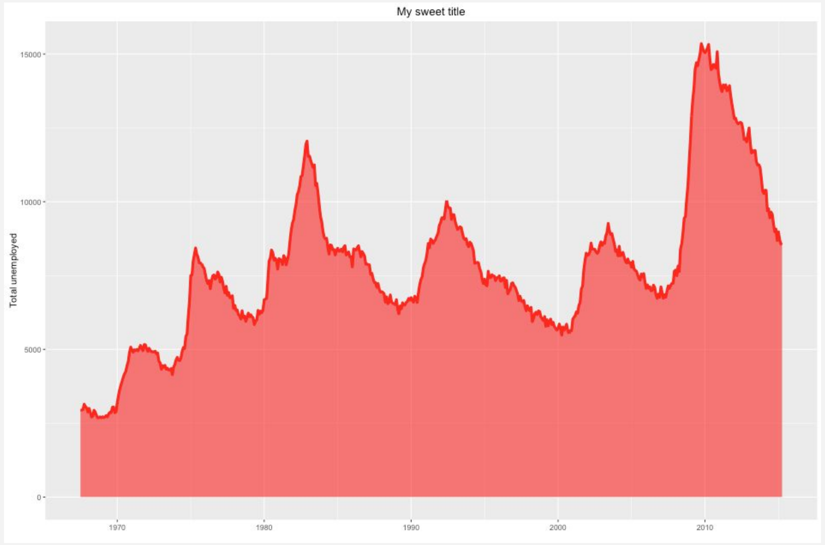
Prettier charts: Opacity, line thickness

```
chart <- ggplot(economics, aes(x=date, y=unemploy)) +
    geom_area(fill="red", alpha=0.6) +
    geom_line(color="red", size=1.5)
```



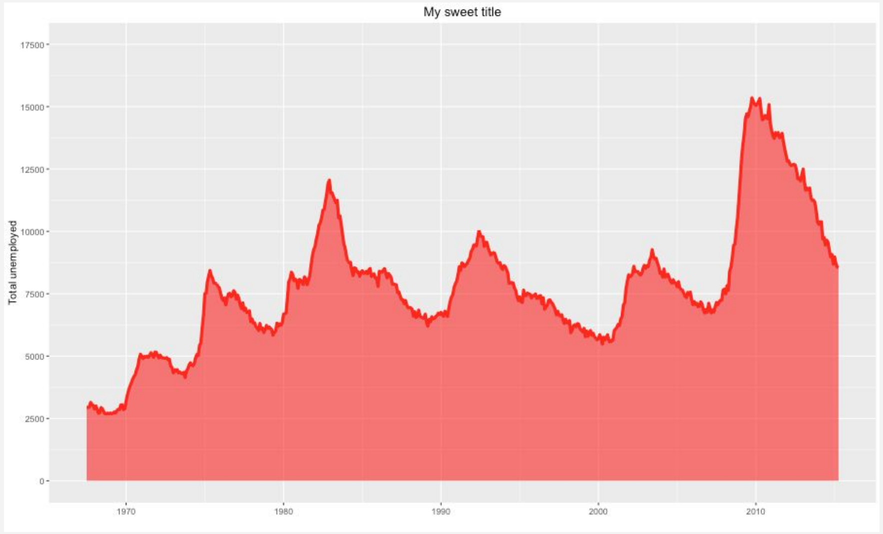
Prettier charts: Labels

```
chart <- ggplot(economics, aes(x=date, y=unemploy)) +
geom_area(fill="red", alpha=0.6) +
geom_line(color="red", size=1.5) +
xlab("") +
ylab("Total unemployed") +
ggtitle("My sweet title")
```



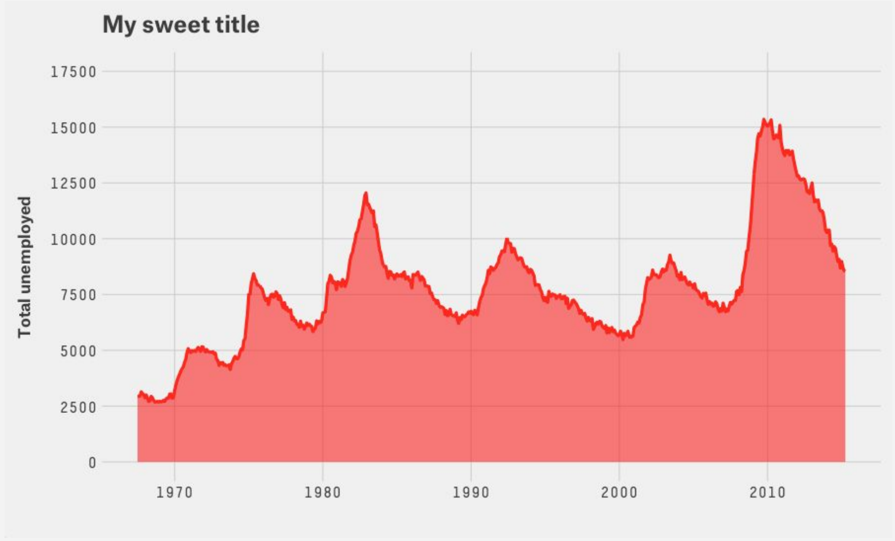
Prettier charts: Custom scales

```
chart <- ggplot(economics, aes(x=date, y=unemploy)) +
geom_area(fill="red", alpha=0.6) +
geom_line(color="red", size=1.5) +
xlab("") +
ylab("Total unemployed") +
ggtitle("My sweet title") +
scale_y_continuous(
   limits=c(0, 17500),
   breaks=seq(0, 20000, 2500)
)
```



Prettier charts: Theme

```
chart <- ggplot(economics, aes(x=date, y=unemploy)) +
geom_area(fill="red", alpha=0.6) +
geom_line(color="red", size=1.5) +
xlab("") +
ylab("Total unemployed") +
ggtitle("My sweet title") +
scale_y_continuous(
    limits=c(0, 17500),
    breaks=seq(0, 20000, 2500)
) +
Five38Thm
```



Installing our theme!

Details here, fonts can take a bit

`https://github.com/fivethirtyeight/theme538`

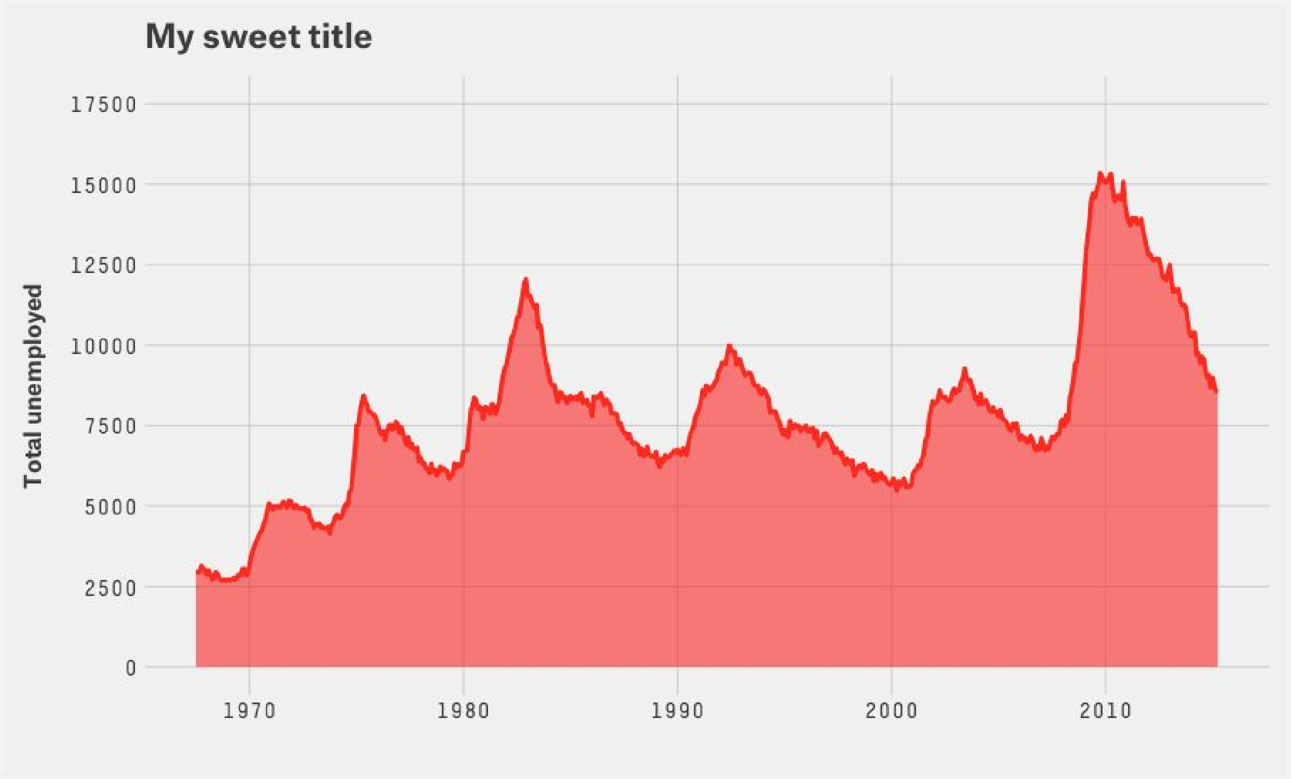
Exporting

```
ggsave("my_chart.pdf", chart, width=10, height=6)
```

When you file a chart using ggplot, send the graphics desk your **data**, **code**, and **output pdf**

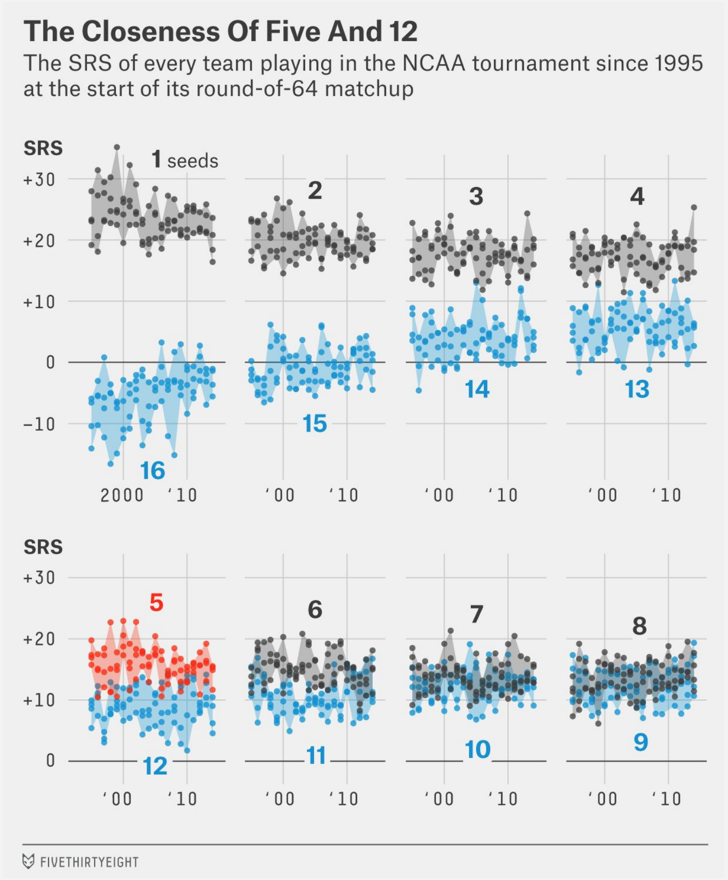
So what?

Lots of code to make a simple Chartbuilder (or Excel) chart



What makes ggplot great?

- Non-standard chart types
- Secondary data encoding
- Grouping and faceting

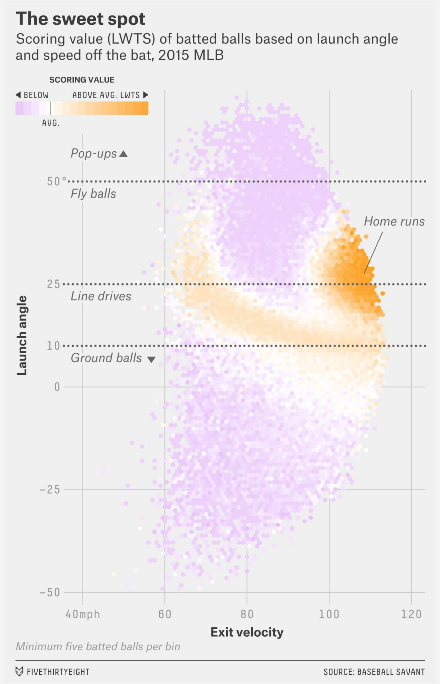


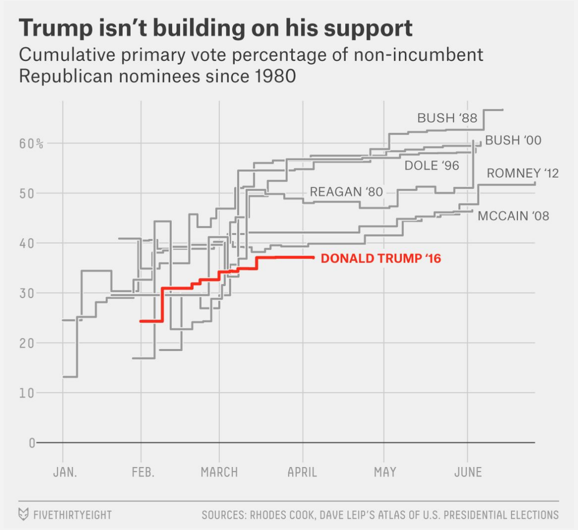
Non-standard charts types

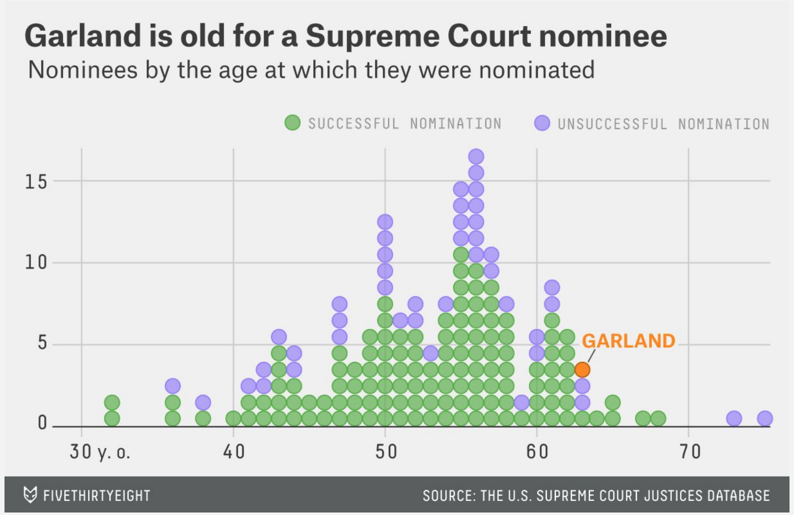
`geom_hex()`

`geom_step()`

`geom_dotplot(method="histodot")`

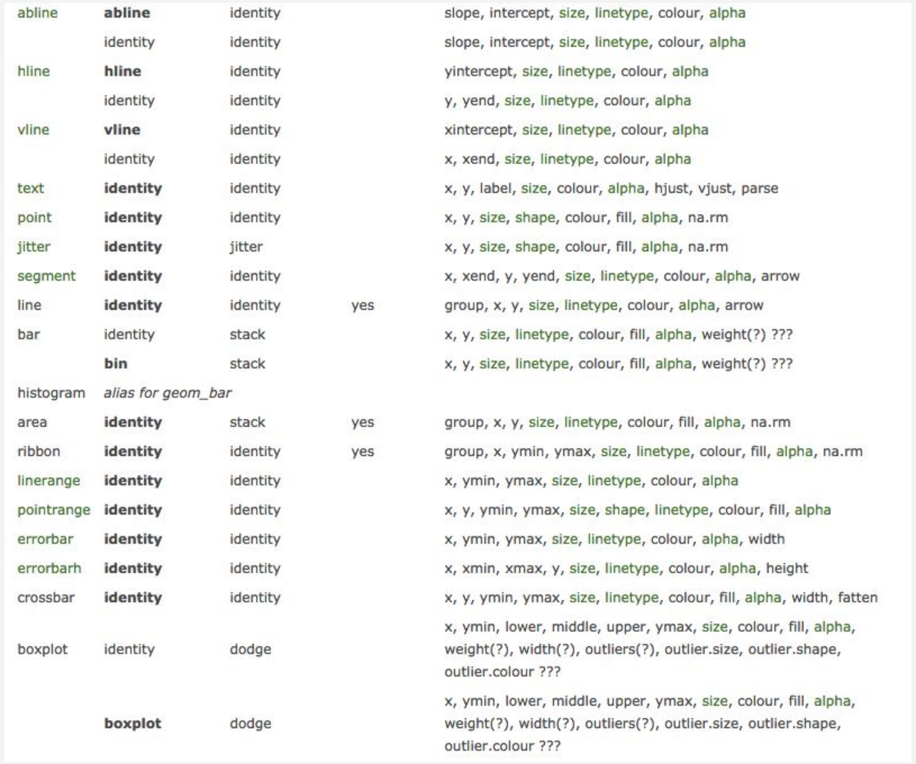




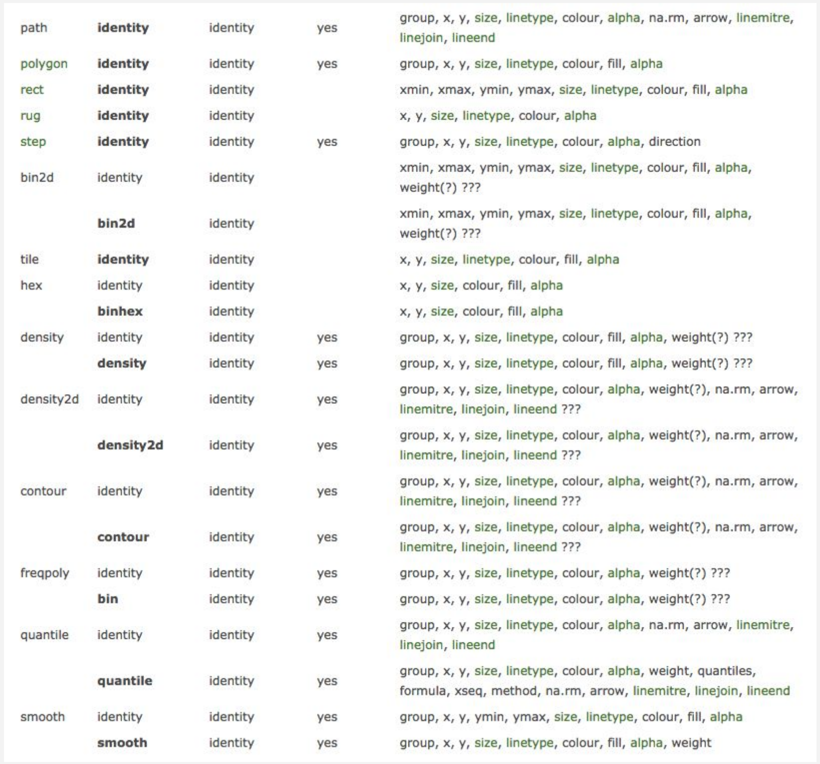


So many options!

Well, not that many options



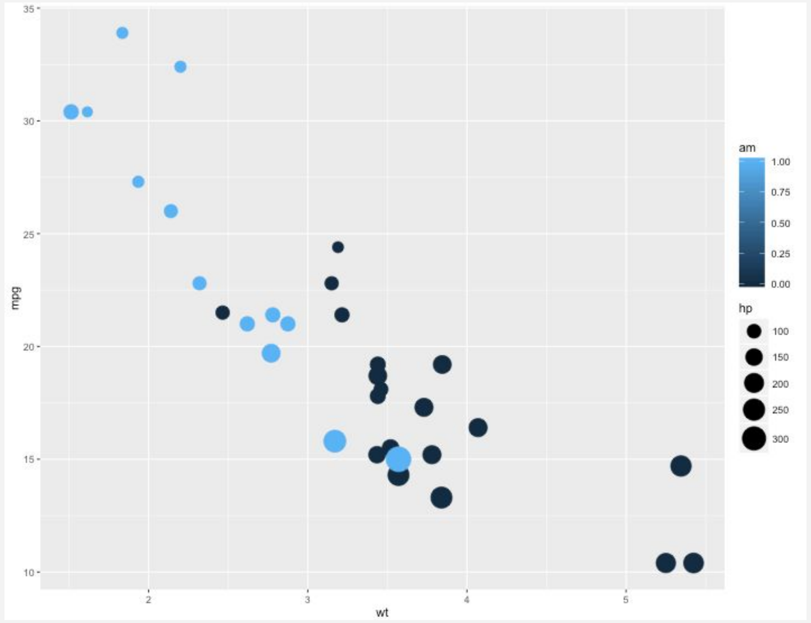
abline	**abline**	identity		slope, intercept, size, linetype, colour, alpha
	identity	identity		slope, intercept, size, linetype, colour, alpha
hline	**hline**	identity		yintercept, size, linetype, colour, alpha
	identity	identity		y, yend, size, linetype, colour, alpha
vline	**vline**	identity		xintercept, size, linetype, colour, alpha
	identity	identity		x, xend, size, linetype, colour, alpha
text	**identity**	identity		x, y, label, size, colour, alpha, hjust, vjust, parse
point	**identity**	identity		x, y, size, shape, colour, fill, alpha, na.rm
jitter	**identity**	jitter		x, y, size, shape, colour, fill, alpha, na.rm
segment	**identity**	identity		x, xend, y, yend, size, linetype, colour, alpha, arrow
line	**identity**	identity	yes	group, x, y, size, linetype, colour, alpha, arrow
bar	identity	stack		x, y, size, linetype, colour, fill, alpha, weight(?) ???
	bin	stack		x, y, size, linetype, colour, fill, alpha, weight(?) ???
histogram	*alias for geom_bar*			
area	**identity**	stack	yes	group, x, y, size, linetype, colour, fill, alpha, na.rm
ribbon	**identity**	identity	yes	group, x, ymin, ymax, size, linetype, colour, fill, alpha, na.rm
linerange	**identity**	identity		x, ymin, ymax, size, linetype, colour, alpha
pointrange	**identity**	identity		x, y, ymin, ymax, size, shape, linetype, colour, fill, alpha
errorbar	**identity**	identity		x, ymin, ymax, size, linetype, colour, alpha, width
errorbarh	**identity**	identity		x, xmin, xmax, y, size, linetype, colour, alpha, height
crossbar	**identity**	identity		x, y, ymin, ymax, size, linetype, colour, fill, alpha, width, fatten
boxplot	identity	dodge		x, ymin, lower, middle, upper, ymax, size, colour, fill, alpha, weight(?), width(?), outliers(?), outlier.size, outlier.shape, outlier.colour ???
	boxplot	dodge		x, ymin, lower, middle, upper, ymax, size, colour, fill, alpha, weight(?), width(?), outliers(?), outlier.size, outlier.shape, outlier.colour ???



path	**identity**	identity	yes	group, x, y, size, linetype, colour, alpha, na.rm, arrow, linemitre, linejoin, lineend
polygon	**identity**	identity	yes	group, x, y, size, linetype, colour, fill, alpha
rect	**identity**	identity		xmin, xmax, ymin, ymax, size, linetype, colour, fill, alpha
rug	**identity**	identity		x, y, size, linetype, colour, alpha
step	**identity**	identity	yes	group, x, y, size, linetype, colour, alpha, direction
bin2d	identity	identity		xmin, xmax, ymin, ymax, size, linetype, colour, fill, alpha, weight(?) ???
	bin2d	identity		xmin, xmax, ymin, ymax, size, linetype, colour, fill, alpha, weight(?) ???
tile	**identity**	identity		x, y, size, linetype, colour, fill, alpha
hex	identity	identity		x, y, size, colour, fill, alpha
	binhex	identity		x, y, size, colour, fill, alpha
density	identity	identity	yes	group, x, y, size, linetype, colour, fill, alpha, weight(?) ???
	density	identity	yes	group, x, y, size, linetype, colour, fill, alpha, weight(?) ???
density2d	identity	identity	yes	group, x, y, size, linetype, colour, alpha, weight(?), na.rm, arrow, linemitre, linejoin, lineend ???
	density2d	identity	yes	group, x, y, size, linetype, colour, alpha, weight(?), na.rm, arrow, linemitre, linejoin, lineend ???
contour	identity	identity	yes	group, x, y, size, linetype, colour, alpha, weight(?), na.rm, arrow, linemitre, linejoin, lineend ???
	contour	identity	yes	group, x, y, size, linetype, colour, alpha, weight(?), na.rm, arrow, linemitre, linejoin, lineend ???
freqpoly	identity	identity	yes	group, x, y, size, linetype, colour, alpha, weight(?) ???
	bin	identity	yes	group, x, y, size, linetype, colour, alpha, weight(?) ???
quantile	identity	identity	yes	group, x, y, size, linetype, colour, alpha, na.rm, arrow, linemitre, linejoin, lineend
	quantile	identity	yes	group, x, y, size, linetype, colour, alpha, weight, quantiles, formula, xseq, method, na.rm, arrow, linemitre, linejoin, lineend
smooth	identity	identity	yes	group, x, y, ymin, ymax, size, linetype, colour, fill, alpha
	smooth	identity	yes	group, x, y, size, linetype, colour, fill, alpha, weight

Secondary data encoding

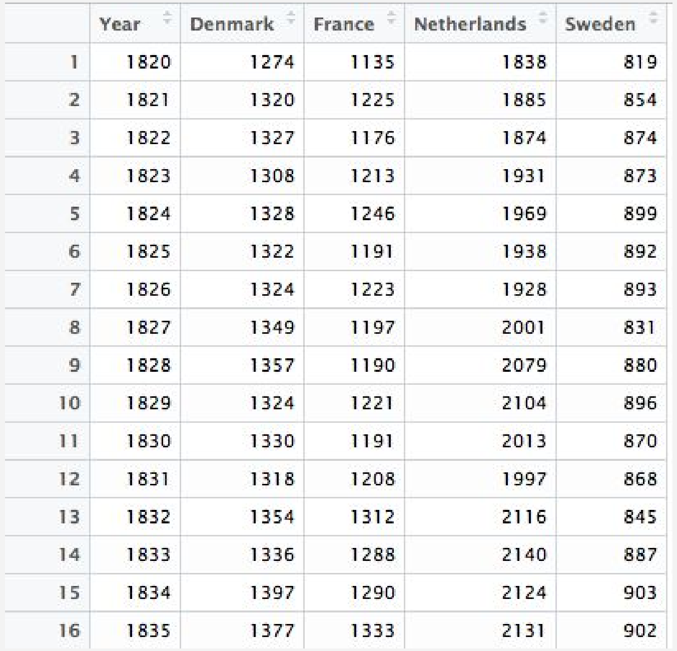
```
chart <- ggplot(mtcars, aes(x=wt, y=mpg, size=hp, color=am))+
    geom_point() +
    scale_size_area(max_size=10)

print(chart)
```



Grouping, faceting and stacking

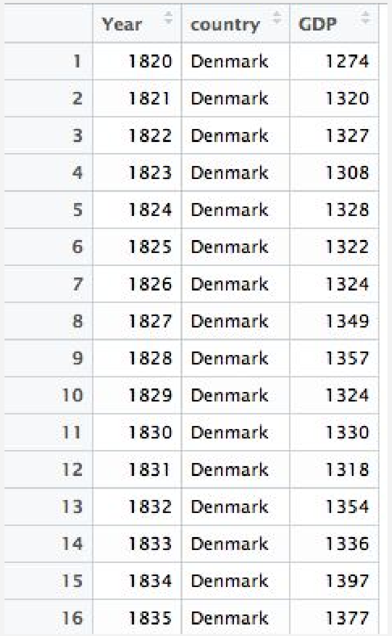
ggplot can do powerful stuff with "long data"

"Wide data" (189 rows)



	Year	Denmark	France	Netherlands	Sweden
1	1820	1274	1135	1838	819
2	1821	1320	1225	1885	854
3	1822	1327	1176	1874	874
4	1823	1308	1213	1931	873
5	1824	1328	1246	1969	899
6	1825	1322	1191	1938	892
7	1826	1324	1223	1928	893
8	1827	1349	1197	2001	831
9	1828	1357	1190	2079	880
10	1829	1324	1221	2104	896
11	1830	1330	1191	2013	870
12	1831	1318	1208	1997	868
13	1832	1354	1312	2116	845
14	1833	1336	1288	2140	887
15	1834	1397	1290	2124	903
16	1835	1377	1333	2131	902

"Long data", after `tidyr` (756 rows)

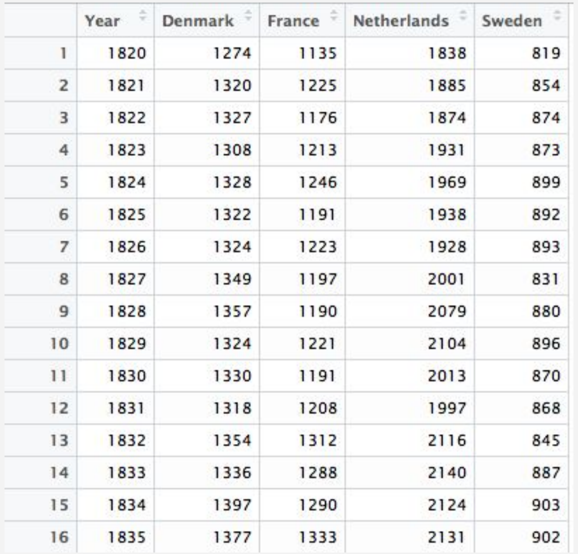


	Year	country	GDP
1	1820	Denmark	1274
2	1821	Denmark	1320
3	1822	Denmark	1327
4	1823	Denmark	1308
5	1824	Denmark	1328
6	1825	Denmark	1322
7	1826	Denmark	1324
8	1827	Denmark	1349
9	1828	Denmark	1357
10	1829	Denmark	1324
11	1830	Denmark	1330
12	1831	Denmark	1318
13	1832	Denmark	1354
14	1833	Denmark	1336
15	1834	Denmark	1397
16	1835	Denmark	1377

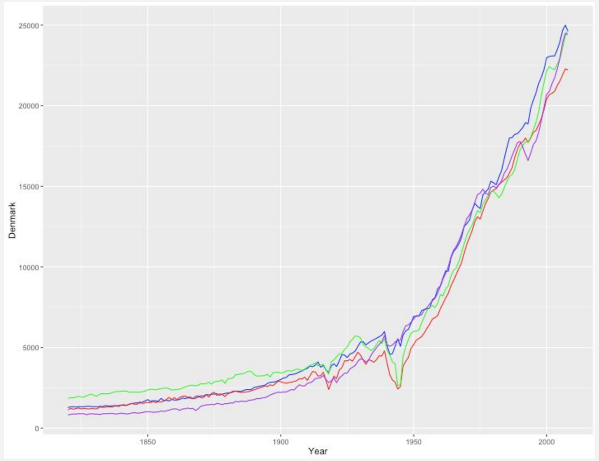
Charting with wide data

Basically how Excel works. Tedious.

```
chart <- ggplot(GDP, aes(x=Year))+
    geom_line(aes(y=Denmark), color='blue')+
    geom_line(aes(y=France), color='red')+
    geom_line(aes(y=Netherlands), color='green')+
    geom_line(aes(y=Sweden), color='purple')

print(chart)
```

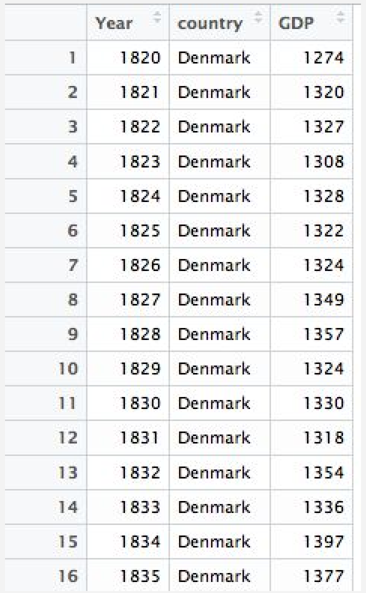


	Year	Denmark	France	Netherlands	Sweden
1	1820	1274	1135	1838	819
2	1821	1320	1225	1885	854
3	1822	1327	1176	1874	874
4	1823	1308	1213	1931	873
5	1824	1328	1246	1969	899
6	1825	1322	1191	1938	892
7	1826	1324	1223	1928	893
8	1827	1349	1197	2001	831
9	1828	1357	1190	2079	880
10	1829	1324	1221	2104	896
11	1830	1330	1191	2013	870
12	1831	1318	1208	1997	868
13	1832	1354	1312	2116	845
14	1833	1336	1288	2140	887
15	1834	1397	1290	2124	903
16	1835	1377	1333	2131	902

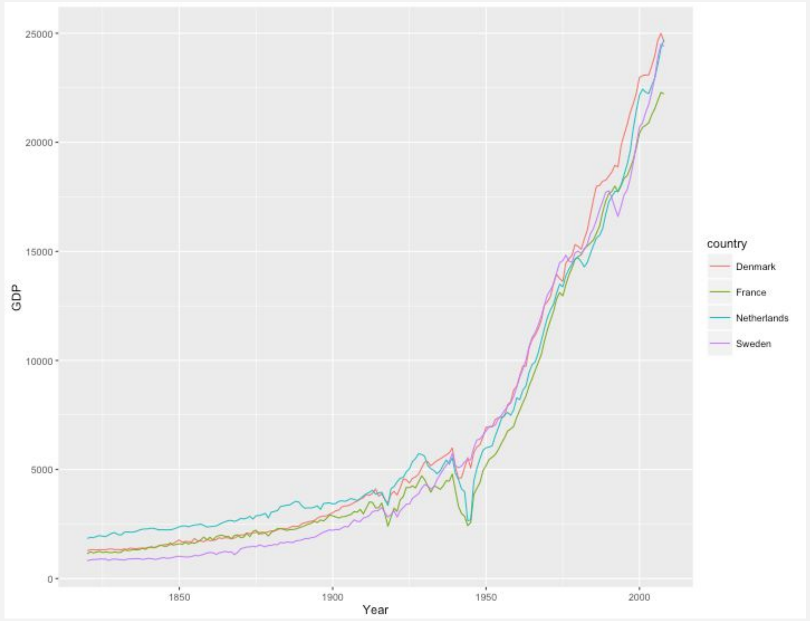


Long data + the group and color aesthetics

```
chart <- ggplot(GDP, aes(x=Year, y=GDP, group=country, color=country))+
    geom_line()

print(chart)
```

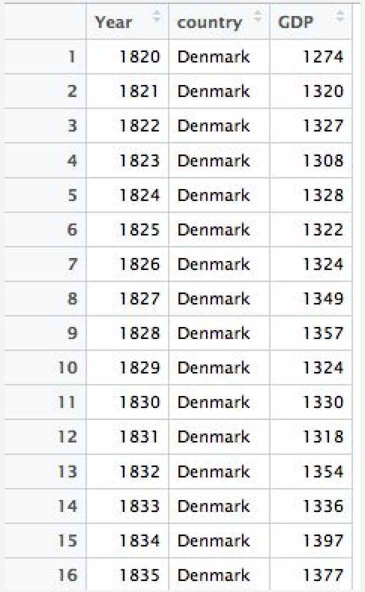


	Year	country	GDP
1	1820	Denmark	1274
2	1821	Denmark	1320
3	1822	Denmark	1327
4	1823	Denmark	1308
5	1824	Denmark	1328
6	1825	Denmark	1322
7	1826	Denmark	1324
8	1827	Denmark	1349
9	1828	Denmark	1357
10	1829	Denmark	1324
11	1830	Denmark	1330
12	1831	Denmark	1318
13	1832	Denmark	1354
14	1833	Denmark	1336
15	1834	Denmark	1397
16	1835	Denmark	1377

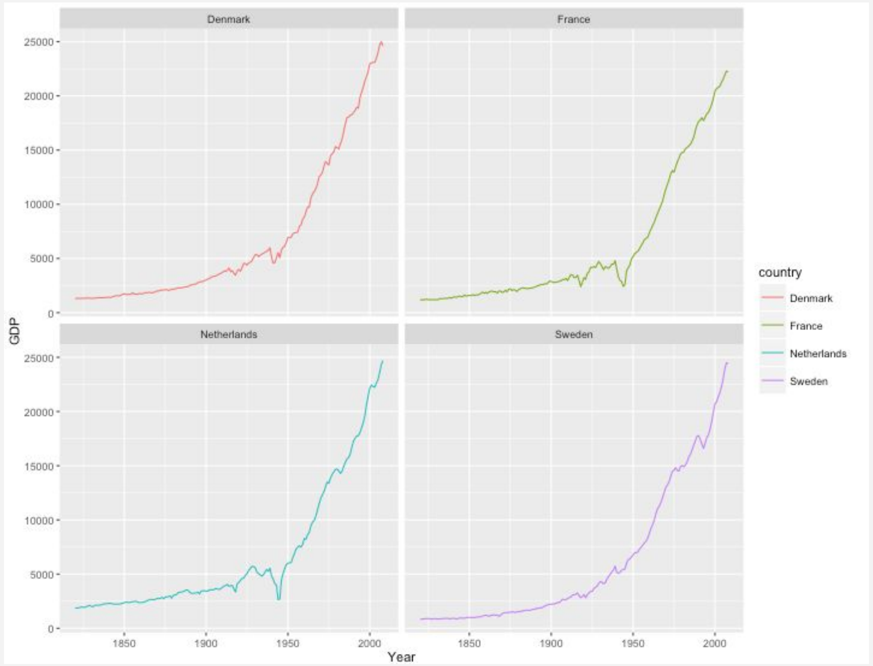


The incredible power of `facet_wrap()`

```
chart <- ggplot(GDP, aes(x=Year, y=GDP, color=country))+
    geom_line()+
    facet_wrap(~country)

print(chart)
```

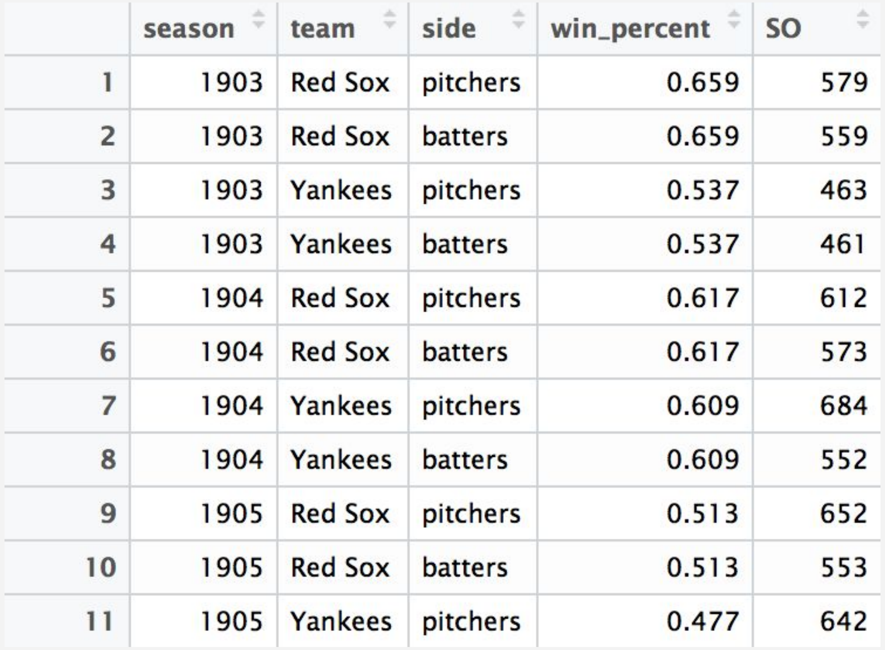


	Year	country	GDP
1	1820	Denmark	1274
2	1821	Denmark	1320
3	1822	Denmark	1327
4	1823	Denmark	1308
5	1824	Denmark	1328
6	1825	Denmark	1322
7	1826	Denmark	1324
8	1827	Denmark	1349
9	1828	Denmark	1357
10	1829	Denmark	1324
11	1830	Denmark	1330
12	1831	Denmark	1318
13	1832	Denmark	1354
14	1833	Denmark	1336
15	1834	Denmark	1397
16	1835	Denmark	1377



Let's make a ~unnecessarily crazy~ chart!

1. Grab the dataset `strikeouts.csv` from the slack channel, and get it into R



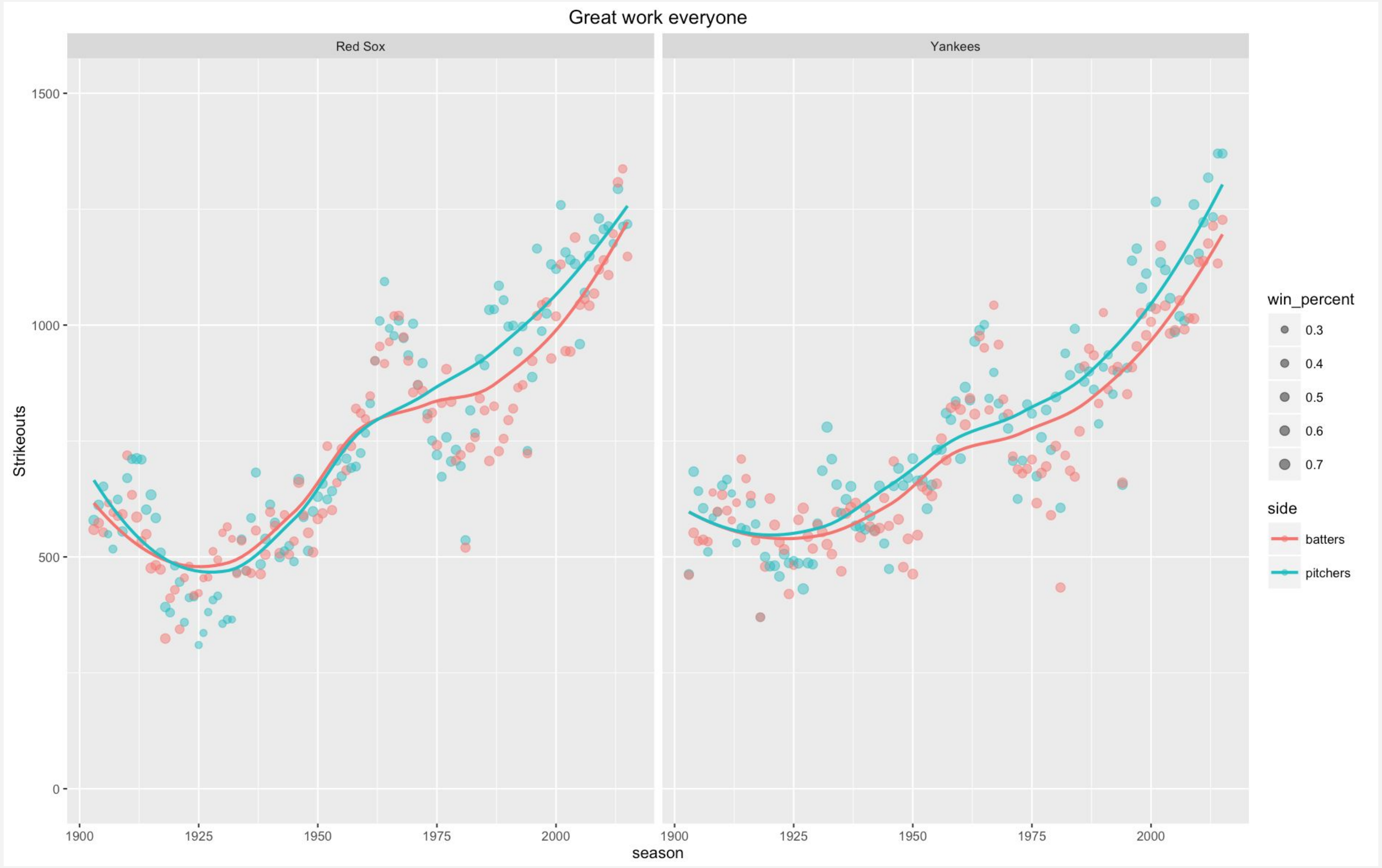
	season	team	side	win_percent	SO
1	1903	Red Sox	pitchers	0.659	579
2	1903	Red Sox	batters	0.659	559
3	1903	Yankees	pitchers	0.537	463
4	1903	Yankees	batters	0.537	461
5	1904	Red Sox	pitchers	0.617	612
6	1904	Red Sox	batters	0.617	573
7	1904	Yankees	pitchers	0.609	684
8	1904	Yankees	batters	0.609	552
9	1905	Red Sox	pitchers	0.513	652
10	1905	Red Sox	batters	0.513	553
11	1905	Yankees	pitchers	0.477	642

Let's make a ~unnecessarily crazy~ chart!

2. Make a chart that:
   a. Shows `season` and against `SO` as a scatterplot
   b. Groups and colors by `side` (pitchers vs. batters)
   c. Facets by `team`
   d. Adds a LOESS smooth (tip: use `se=FALSE` to get rid of gray error range)
   e. Gives each point an opacity of 50%
   f. Sizes the points by `win_percent` (for extra credit put the `aes` call in a spot where it only applies to points, not the LOESS)
   g. Has a proper y-axis label and title
   h. Scales the y-axis so it goes from 0 to 1500

Stuff we didn't cover today

- Labeling data points (`geom_text`)
- Highlighting specific parts of the data
- Reordering factor levels — single worst part of ggplot
- Useful coordinate systems (`coord_flip` and `coord_fixed`)
- Color scales
- Bar charts/stacking (`geom_bar`)
- Histograms/binning (`geom_histogram`)
- Mapping


Great work everyone
Red Sox
Yankees
Strikeouts
1500
1000
500
0
1900
1925
1950
1975
2000
season
win_percent
0.3
0.4
0.5
0.6
0.7
side
batters
pitchers