Reshaping data and data visualization

Frank Edwards

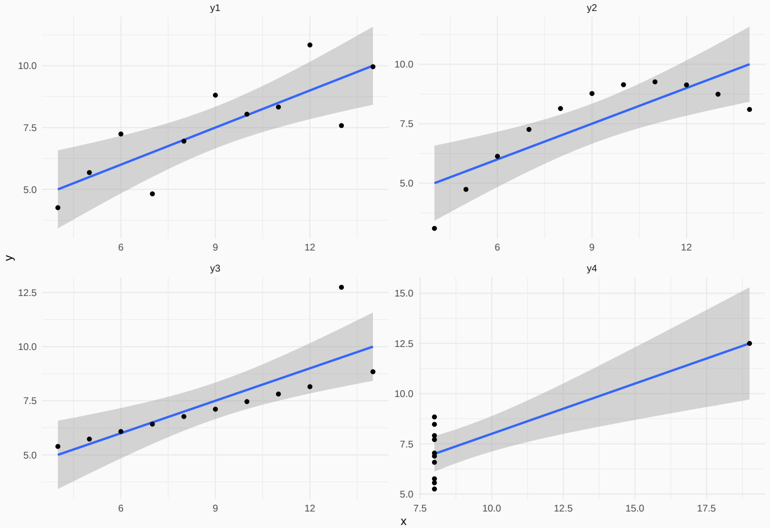
2024-02-06

Review HW 3

Data visualization

- What makes a good visual?
- Why visualize?
- How to use ggplot to make visuals in R

Why do we visualize data?

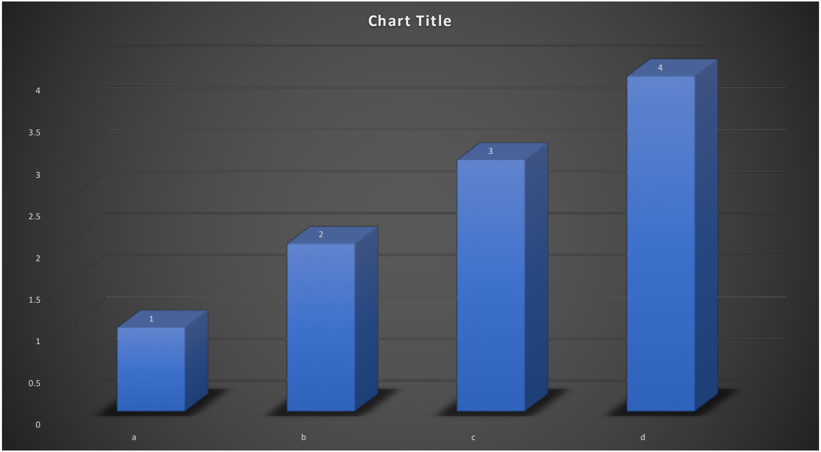


Principles of good data visuals

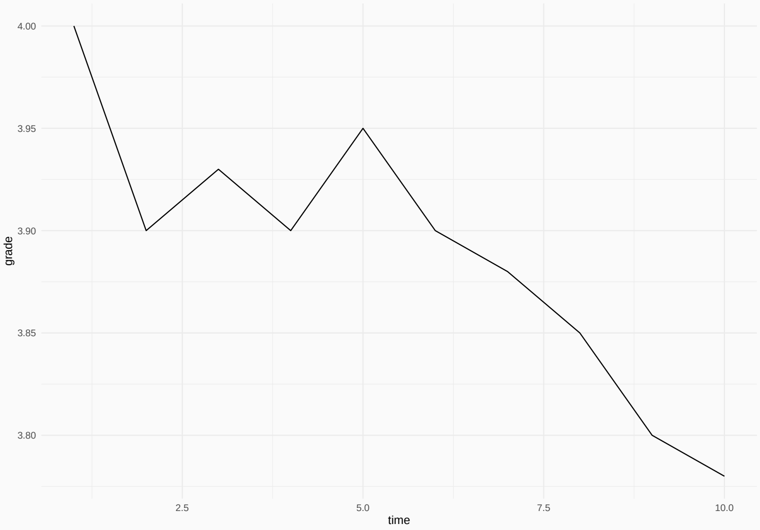
Good data visuals

- Are clearly labeled
- Avoid deception
- Use repetition to invite comparisons
- Minimize ‘chartjunk’

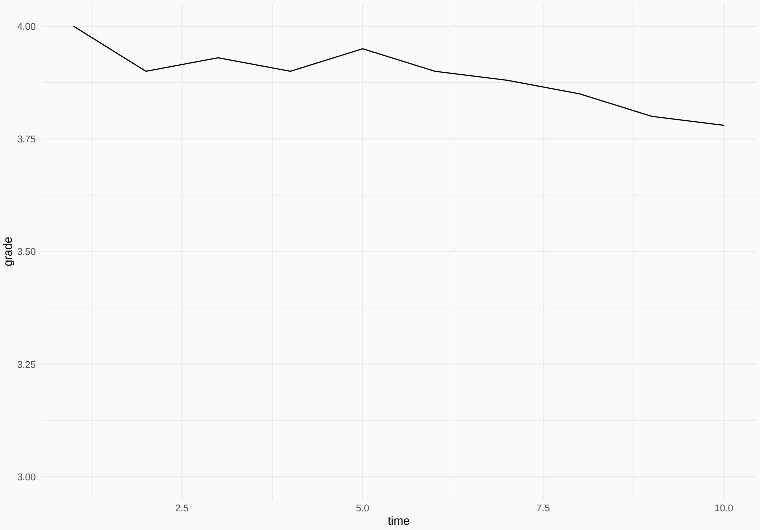
Find the chartjunk



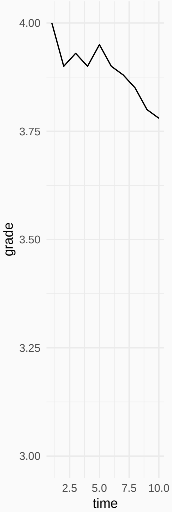
The importance of axes



The importance of axes



The importance of aspect ratio



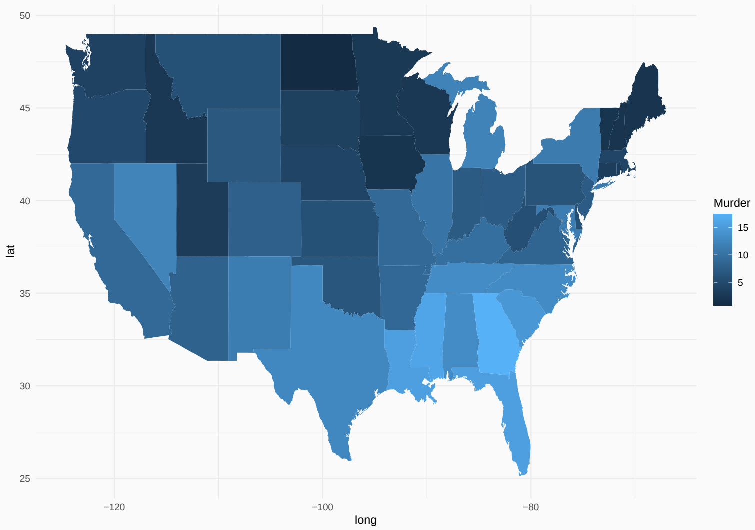
Why Visualize Data?

Why do we visualize data?

- Visuals can quickly reveal patterns in data
- Visuals are a (more) effective way to communicate quantitative information

Geographic Data

	V1	V2
1	Alabama	13.2
2	Alaska	10
3	Arizona	8.1
4	Arkansas	8.8
5	California	9
6	Colorado	7.9
7	Connecticut	3.3
8	Delaware	5.9
9	Florida	15.4
10	Georgia	17.4
11	Hawaii	5.3
12	Idaho	2.6
13	Illinois	10.4
14	Indiana	7.2
15	Iowa	2.2
16	Kansas	6
17	Kentucky	9.7
18	Louisiana	15.4
19	Maine	2.1
20	Maryland	11.3
21	Massachusetts	4.4
22	Michigan	12.1
23	Minnesota	2.7
24	Mississippi	16.1
25	Missouri	9
26	Montana	6
27	Nebraska	4.3
28	Nevada	12.2
29	New Hampshire	2.1
30	New Jersey	7.4
31	New Mexico	11.4
32	New York	11.1

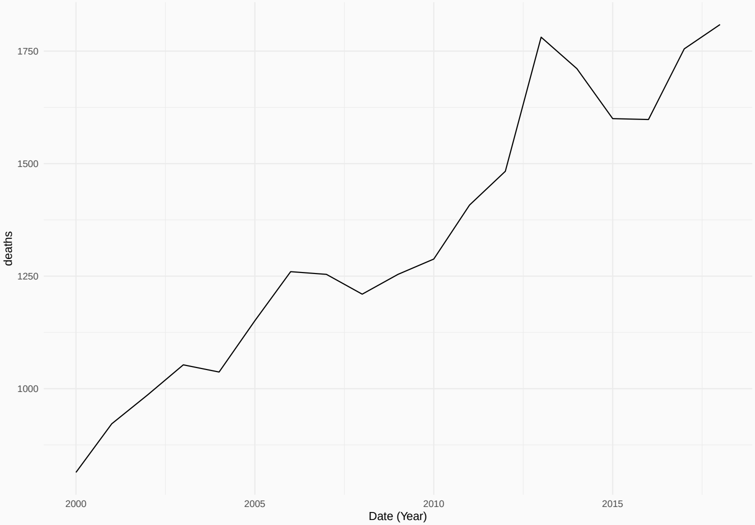


50
45
40
35
30
25
lat
−120
−100
−80
long
Murder
15
10
5

Which is most effective? Why?

Time Series

	Date (Year)	deaths
1	2000	814
2	2001	922
3	2002	986
4	2003	1053
5	2004	1037
6	2005	1151
7	2006	1260
8	2007	1254
9	2008	1210
10	2009	1254
11	2010	1288
12	2011	1408
13	2012	1483
14	2013	1781
15	2014	1711
16	2015	1600
17	2016	1598
18	2017	1755
19	2018	1809


1750
1500
1250
1000
deaths
2000
2005
2010
2015
Date (Year)

Which is most effective? Why?

Model results

Investigated police child maltreatment reports, parameter estimates and standard errors for multilevel poisson regression. Results combined across multiple imputations

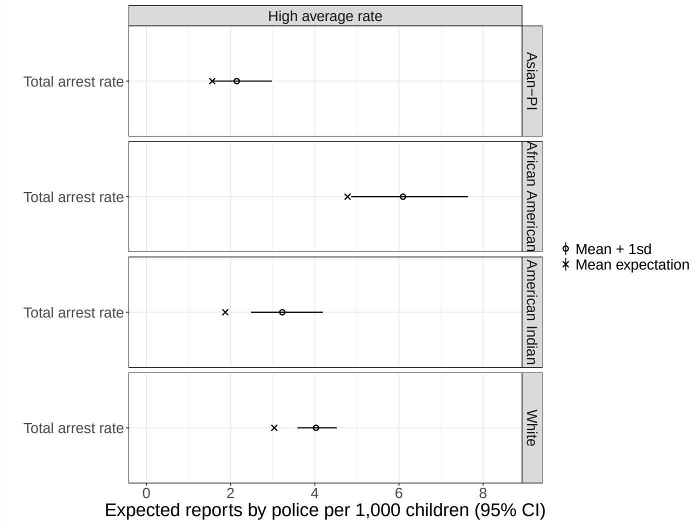
	All arrests	Violent arrests	Drug arrests	QoL arrests
Intercept	-5.80***	-5.74***	-5.77***	-5.61***
	(0.04)	(0.04)	(0.04)	(0.04)
Asian Am/PI	-0.66***	-0.83***	-0.79***	-0.94***
	(0.06)	(0.07)	(0.06)	(0.07)
Native Am	-0.48***	-0.56***	-0.26***	-0.79***
	(0.05)	(0.06)	(0.05)	(0.06)
African Am	0.45***	0.42***	0.43***	0.36***
	(0.04)	(0.04)	(0.04)	(0.04)
Mean arrest	0.28***	0.29***	0.25***	0.15***
	(0.02)	(0.02)	(0.02)	(0.01)
Change in arrest	0.03***	0.01***	0.02***	0.02***
	(0.01)	(0.01)	(0.01)	(0.01)
Mean child pov	0.30***	0.30***	0.31***	0.34***
	(0.02)	(0.02)	(0.02)	(0.02)
Change in child pov	0.00	0.00	0.00	0.00
	(0.01)	(0.01)	(0.01)	(0.01)
Year	0.09***	0.08***	0.08***	0.09***
	(0.00)	(0.00)	(0.01)	(0.00)
No. of police depts	0.05***	0.04***	0.04***	0.04***
	(0.01)	(0.01)	(0.01)	(0.01)
UR	0.07	0.11	0.09	0.07
	(0.04)	(0.04)	(0.05)	(0.04)
UR1	-0.04	-0.09	-0.07	-0.09
	(0.04)	(0.04)	(0.04)	(0.04)
UR2	0.01	0.01	-0.01	0.02
	(0.03)	(0.03)	(0.03)	(0.03)
UR3	-0.03	-0.07	-0.03	-0.06
	(0.03)	(0.03)	(0.03)	(0.03)
UR4	0.02	0.03	0.03	0.03
	(0.02)	(0.03)	(0.03)	(0.03)
Officers per cap	-0.03*	-0.02*	-0.02*	-0.01*
	(0.01)	(0.01)	(0.01)	(0.01)
Pct pop	0.40***	0.34***	0.35***	0.20***
	(0.04)	(0.04)	(0.04)	(0.04)
Asian Am/PI x Mean arrest	0.03	-0.04	-0.00	0.09
	(0.04)	(0.03)	(0.04)	(0.03)
Native Am x Mean arrest	0.26***	0.27***	0.35***	0.23***
	(0.02)	(0.02)	(0.02)	(0.02)
African Am x Mean arrest	-0.04*	-0.11*	-0.03*	0.06*
	(0.02)	(0.02)	(0.02)	(0.02)
Asian Am/PI x change in arrest	0.03	0.02	0.00	0.02
	(0.02)	(0.03)	(0.02)	(0.02)
Native Am x change in arrest	0.01	0.02	0.00	0.01
	(0.02)	(0.02)	(0.02)	(0.01)
African Am x change in arrest	-0.01	0.00	-0.00	-0.00
	(0.01)	(0.01)	(0.01)	(0.01)
Asian Am/PI x Mean child pov	-0.27***	-0.26***	-0.28***	-0.30***
	(0.03)	(0.03)	(0.03)	(0.02)
Native Am x Mean child pov	-0.18***	-0.13***	-0.16***	-0.15***
	(0.03)	(0.03)	(0.03)	(0.03)
African Am x Mean child pov	-0.22***	-0.22***	-0.23***	-0.26***
	(0.02)	(0.02)	(0.02)	(0.02)
Asian Am/PI x Change in child pov	-0.01	-0.01	-0.01	-0.01
	(0.02)	(0.02)	(0.02)	(0.02)
Native Am x Change in child pov	-0.01	0.00	-0.01	-0.00
	(0.02)	(0.02)	(0.02)	(0.02)
African Am x Change in child pov	-0.01	-0.01	-0.01	-0.01
	(0.01)	(0.01)	(0.01)	(0.01)
Asian Am/PI x Pct pop	-0.60***	-0.56***	-0.56***	-0.36***
	(0.07)	(0.07)	(0.07)	(0.07)
Native Am x Pct pop	-0.57***	-0.51***	-0.38***	-0.37***
	(0.05)	(0.05)	(0.05)	(0.05)
African Am x Pct pop	-0.95***	-0.93***	-0.89***	-0.72***
	(0.05)	(0.05)	(0.05)	(0.05)
Residual variance	0.36	0.36	0.36	0.36
County intercept variance	0.19	0.19	0.20	0.20

***p < 0.001, **p < 0.01, *p < 0.05

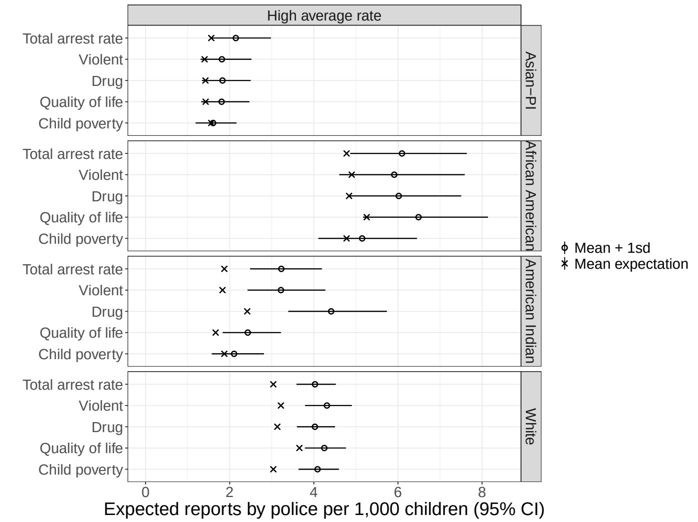
Reduced format: focal variable sign and significance

	Parameter	All	Violent	Drug	Quality of life
Total	Between counties	+	+	+	+
	Within county	+	+	+	+
African American	Between counties	+	+	+	+
	Within county	+	+	+	+
Asian-Pacific Islander	Between counties	+	+	+	+
	Within county	+	+		+
American Indian / Alaska Native	Between counties	+	+	+	+
	Within county	+	+	+	+
White	Between counties	+	+	+	+
	Within county	+	+	+	+

Plot summary



Plot summary



Which is most effective? Why?

Break

Using ggplot2 to visualize data in R

The importance of tidy (long) data for ggplot

Data is generally either wide or long

- In wide format, column position may indicate a variables value
- In long format, each variable has its own column

Example of long data: each column is a variable

```r
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Example of the same data in wide format

```
##   setosa.Sepal.Length setosa.Sepal.Width setosa.Petal.Length setosa.Petal.Width
## 1                 5.1                3.5                 1.4                0.2
## 2                 4.9                3.0                 1.4                0.2
## 3                 4.7                3.2                 1.3                0.2
## 4                 4.6                3.1                 1.5                0.2
## 5                 5.0                3.6                 1.4                0.2
## 6                 5.4                3.9                 1.7                0.4
##   versicolor.Sepal.Length versicolor.Sepal.Width versicolor.Petal.Length
## 1                     7.0                    3.2                     4.7
## 2                     6.4                    3.2                     4.5
## 3                     6.9                    3.1                     4.9
## 4                     5.5                    2.3                     4.0
## 5                     6.5                    2.8                     4.6
## 6                     5.7                    2.8                     4.5
##   versicolor.Petal.Width virginica.Sepal.Length virginica.Sepal.Width
## 1                    1.4                    6.3                   3.3
## 2                    1.5                    5.8                   2.7
## 3                    1.5                    7.1                   3.0
## 4                    1.3                    6.3                   2.9
## 5                    1.5                    6.5                   3.0
## 6                    1.3                    7.6                   3.0
##   virginica.Petal.Length virginica.Petal.Width
## 1                    6.0                   2.5
## 2                    5.1                   1.9
## 3                    5.9                   2.1
## 4                    5.6                   1.8
## 5                    5.8                   2.2
## 6                    6.6                   2.1
```

Tidy data lets us efficiently feed aesthetic parameters to ggplot.

- Tidy data is harder for humans to read in a spreadsheet, but much easier to program with. Tidyverse packages are built around making and keeping our R objects in tidy (long data.frame) format
- Try to keep your data tidy - all variables should be variables, not embedded in column names.

Frequent untidy variables:

- Time (i.e. year)
- Group

Basic anatomy of a ggplot command

Give it data

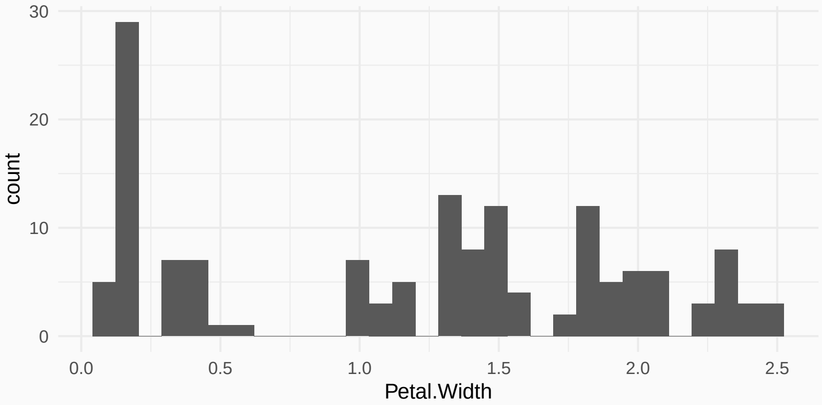
```
data("iris")
my_plot <- ggplot(data = iris)
```

Add a single aesthetic parameter

```r
ggplot(data = iris, aes(x = Petal.Width))
```

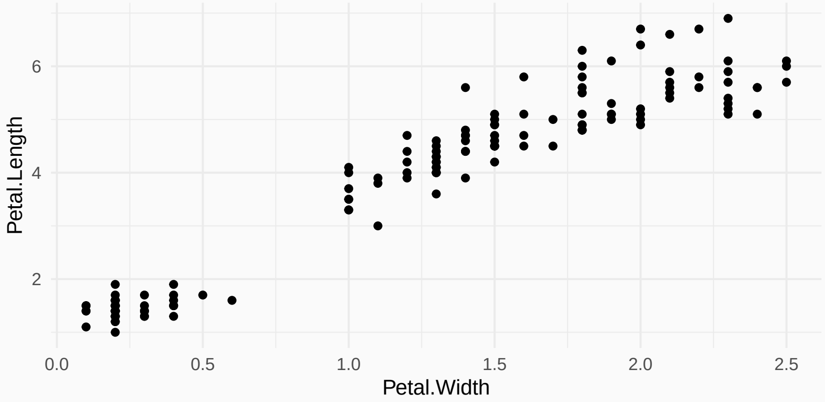
Add a geom

```r
ggplot(data = iris, aes(x = Petal.Width)) + geom_histogram()
```



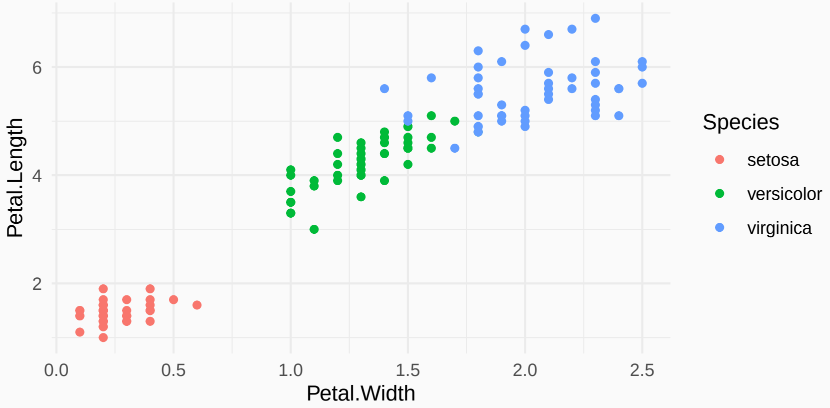
Add two aesthetic parameters and a geom

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) + geom_point()
```



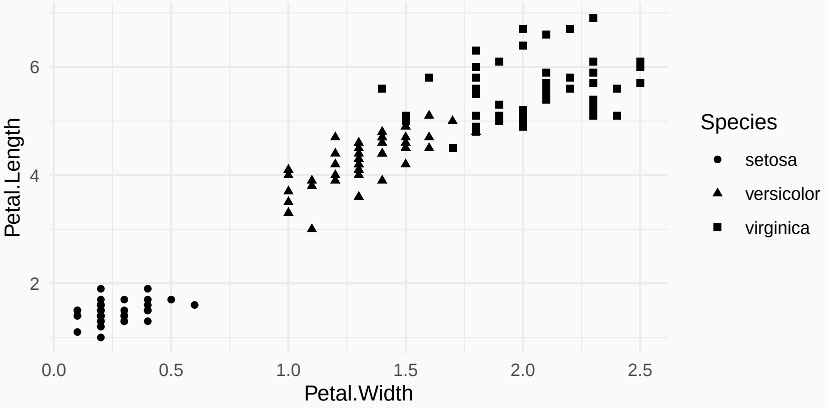
Three variables: two continuous, one categorical

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, color = Species)) + geom
```



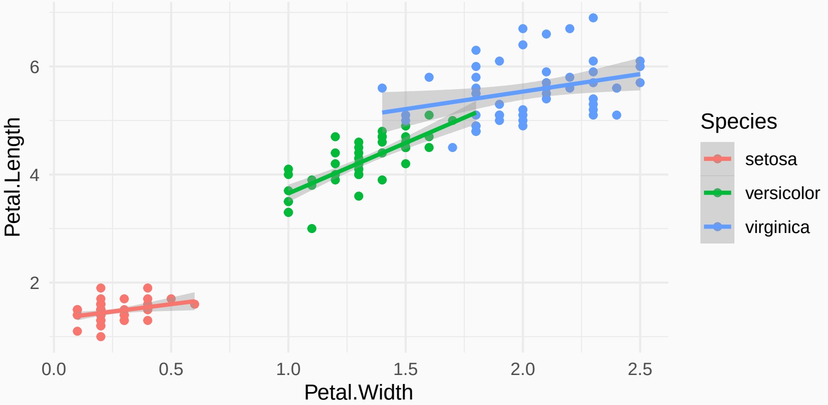
Three variables: two continuous, one categorical

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, shape = Species)) + geom
```



Multiple geoms

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, color = Species)) + geom
    geom_smooth(method = "lm")
```



To review

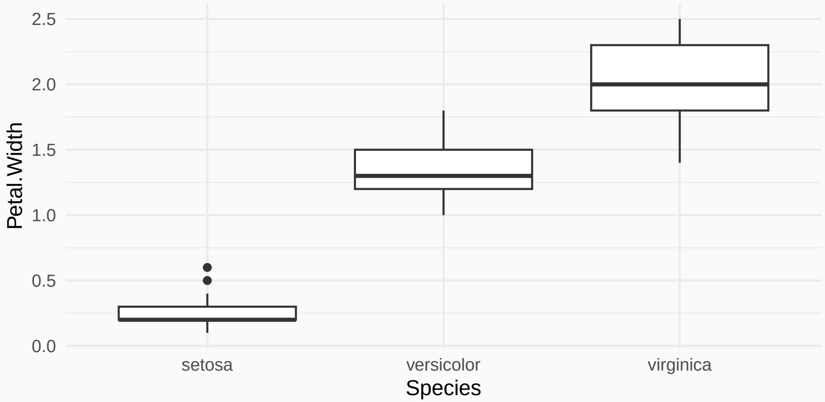
ggplot needs three things to make a graphic

1. Data
2. Aesthetic paramaters
3. Geoms

More advanced plots

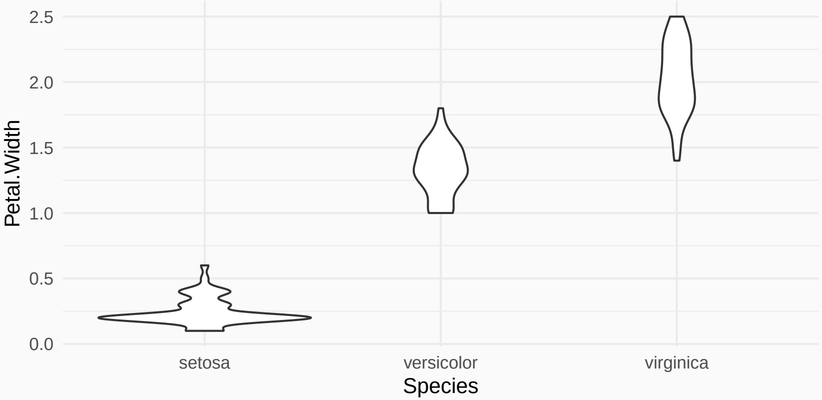
Boxplots (one continuous, one categorical)

```
ggplot(data = iris, aes(y = Petal.Width, x = Species)) + geom_boxplot()
```



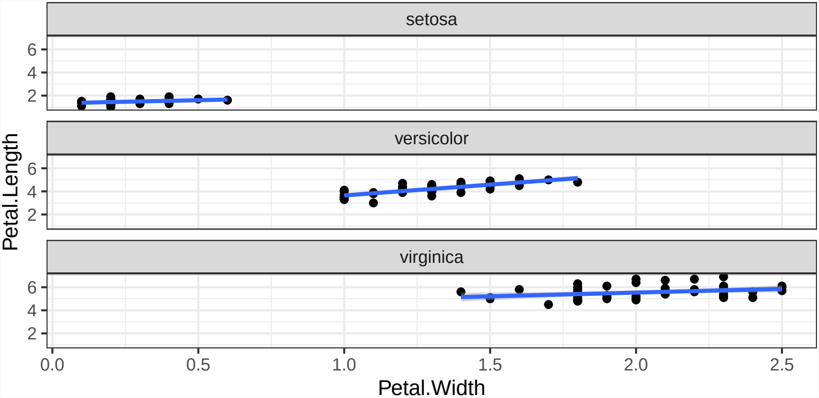
Violin plot

```
ggplot(data = iris, aes(y = Petal.Width, x = Species)) + geom_violin()
```

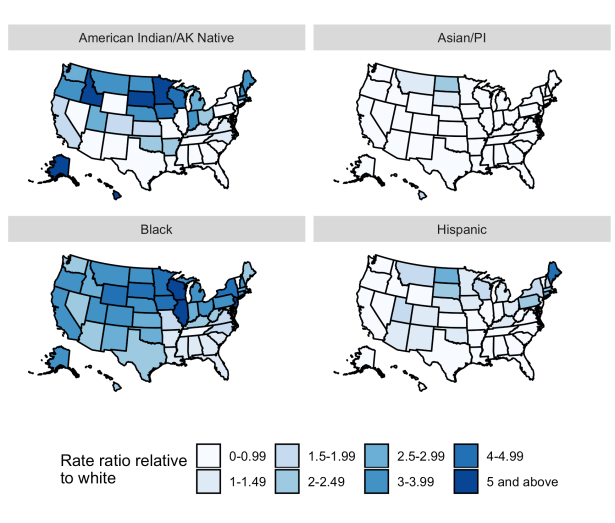


Small multiples (facets)

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) + geom_point() + geom_s
    theme_bw() + facet_wrap(~Species, ncol = 1)
```

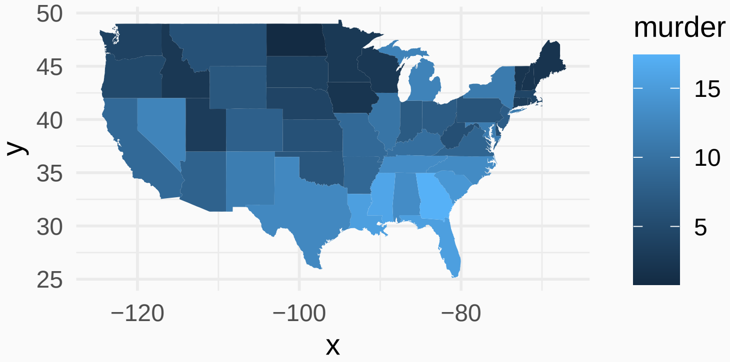


Small multiples are very powerful



Maps

```
data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))
map <- map_data("state")
ggplot(data, aes(fill = murder)) + geom_map(aes(map_id = state), map = map) + expand_limits(x = map$long,
    y = map$lat)
```

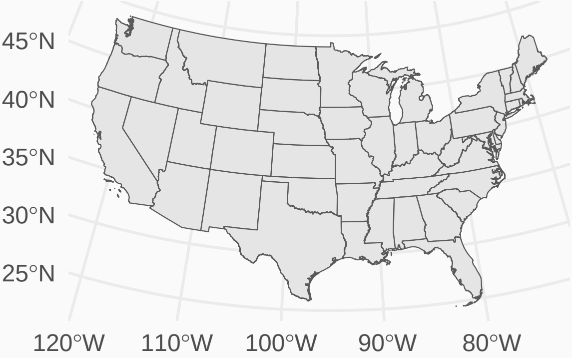


Shape files

```
library(tigris)
library(sf)
st <- states(cb = T) %>%
    st_transform(8528) %>%
    filter(!STUSPS %in% c("HI", "AK", "PR", "GU", "AS", "VI", "MP"))
```

```
##   |                                                                        |
```

```
ggplot(st) + geom_sf()
```



Reshaping data using the tidyverse: Grouping and summarizing

Evaluating the structure of the data

```r
library(gapminder)
head(gapminder)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
## 5 Afghanistan Asia       1972    36.1 13079460      740.
## 6 Afghanistan Asia       1977    38.4 14880372      786.
```

How is this data structured?

What natural groupings are present in this data?

Grouping and summarizing: by country

```
gapminder %>%
    group_by(country) %>%
    summarise(mean_lifeExp = mean(lifeExp))
```

```
## # A tibble: 142 x 2
##    country     mean_lifeExp
##    <fct>              <dbl>
##  1 Afghanistan         37.5
##  2 Albania             68.4
##  3 Algeria             59.0
##  4 Angola              37.9
##  5 Argentina           69.1
##  6 Australia           74.7
##  7 Austria             73.1
##  8 Bahrain             65.6
##  9 Bangladesh          49.8
## 10 Belgium             73.6
## # i 132 more rows
```

Grouping and summarizing: by country (cont.)

```
gapminder %>%
    group_by(country) %>%
    summarise(mean_lifeExp = mean(lifeExp), max_lifeExp = max(lifeExp), min_lifeExp = min(lifeExp))
```

```
## # A tibble: 142 x 4
##    country     mean_lifeExp max_lifeExp min_lifeExp
##    <fct>              <dbl>       <dbl>       <dbl>
##  1 Afghanistan         37.5        43.8        28.8
##  2 Albania             68.4        76.4        55.2
##  3 Algeria             59.0        72.3        43.1
##  4 Angola              37.9        42.7        30.0
##  5 Argentina           69.1        75.3        62.5
##  6 Australia           74.7        81.2        69.1
##  7 Austria             73.1        79.8        66.8
##  8 Bahrain             65.6        75.6        50.9
##  9 Bangladesh          49.8        64.1        37.5
## 10 Belgium             73.6        79.4        68
## # i 132 more rows
```

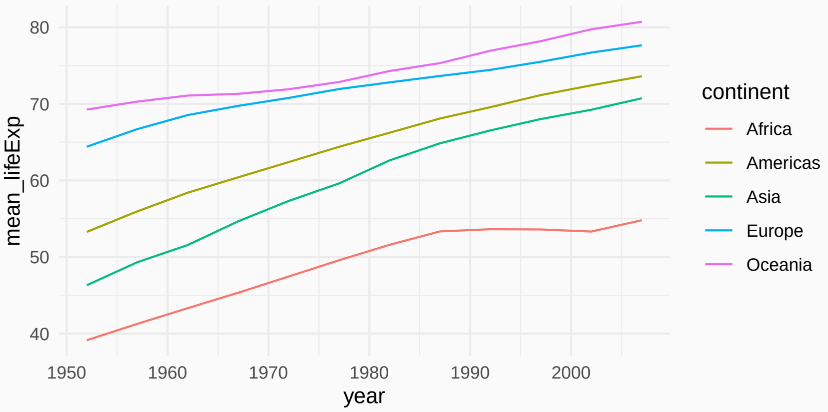
Grouping and summarizing: by continent and year

```
gapminder %>%
    group_by(continent, year) %>%
    summarise(mean_lifeExp = mean(lifeExp))
```

```
## # A tibble: 60 x 3
## # Groups:   continent [5]
##    continent  year mean_lifeExp
##    <fct>     <int>        <dbl>
##  1 Africa     1952         39.1
##  2 Africa     1957         41.3
##  3 Africa     1962         43.3
##  4 Africa     1967         45.3
##  5 Africa     1972         47.5
##  6 Africa     1977         49.6
##  7 Africa     1982         51.6
##  8 Africa     1987         53.3
##  9 Africa     1992         53.6
## 10 Africa     1997         53.6
## # i 50 more rows
```

Grouping and summarizing: by continent and year (cont.)

```r
gapminder %>%
    group_by(continent, year) %>%
    summarise(mean_lifeExp = mean(lifeExp)) %>%
    ggplot(aes(x = year, y = mean_lifeExp, col = continent)) + geom_line()
```



Reshaping with pivots (long<->wide)

Is this data long or wide?

```r
dat <- gapminder %>%
    group_by(continent, year) %>%
    summarise(mean_lifeExp = mean(lifeExp))
head(dat)
```

```
## # A tibble: 6 x 3
## # Groups:   continent [1]
##   continent  year mean_lifeExp
##   <fct>     <int>        <dbl>
## 1 Africa     1952         39.1
## 2 Africa     1957         41.3
## 3 Africa     1962         43.3
## 4 Africa     1967         45.3
## 5 Africa     1972         47.5
## 6 Africa     1977         49.6
```

Use pivot_wider to make it wide by continent

```r
dat_wide <- dat %>%
    pivot_wider(names_from = continent, values_from = mean_lifeExp)
head(dat_wide)
```

```
## # A tibble: 6 x 6
##    year Africa Americas  Asia Europe Oceania
##   <int>  <dbl>    <dbl> <dbl>  <dbl>   <dbl>
## 1  1952   39.1     53.3  46.3   64.4    69.3
## 2  1957   41.3     56.0  49.3   66.7    70.3
## 3  1962   43.3     58.4  51.6   68.5    71.1
## 4  1967   45.3     60.4  54.7   69.7    71.3
## 5  1972   47.5     62.4  57.3   70.8    71.9
## 6  1977   49.6     64.4  59.6   71.9    72.9
```

Use pivot_wider to make it wide by year

```r
dat_wide <- dat %>%
    pivot_wider(names_from = year, values_from = mean_lifeExp)
head(dat_wide)
```

```
## # A tibble: 5 x 13
## # Groups:   continent [5]
##   continent `1952` `1957` `1962` `1967` `1972` `1977` `1982` `1987` `1992`
##   <fct>      <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Africa      39.1   41.3   43.3   45.3   47.5   49.6   51.6   53.3   53.6
## 2 Americas    53.3   56.0   58.4   60.4   62.4   64.4   66.2   68.1   69.6
## 3 Asia        46.3   49.3   51.6   54.7   57.3   59.6   62.6   64.9   66.5
## 4 Europe      64.4   66.7   68.5   69.7   70.8   71.9   72.8   73.6   74.4
## 5 Oceania     69.3   70.3   71.1   71.3   71.9   72.9   74.3   75.3   76.9
## # i 3 more variables: `1997` <dbl>, `2002` <dbl>, `2007` <dbl>
```

Use pivot_longer() to make wide data long

```r
dat_long <- dat_wide %>%
    pivot_longer(cols = "1952":"2007", values_to = "mean_lifeExp")
head(dat_long)
```

```
## # A tibble: 6 x 3
## # Groups:   continent [1]
##   continent name  mean_lifeExp
##   <fct>     <chr>        <dbl>
## 1 Africa    1952          39.1
## 2 Africa    1957          41.3
## 3 Africa    1962          43.3
## 4 Africa    1967          45.3
## 5 Africa    1972          47.5
## 6 Africa    1977          49.6
```