# Data transformations continued and data visualization

# Overview

## Data transformations using dplyr

- Review of the main functions in dplyr
- Using dplyr to compare audience and critic movie ratings

## Data visualization using ggplot

- Conceptual overview of the grammar of graphics
- If there is time:
  - Using ggplot to create data visualizations

# Homework 3

It is due on Gradescope by 11pm on Monday July 21st

How did homework 2 go?

# Data wrangling/transformation using dplyr

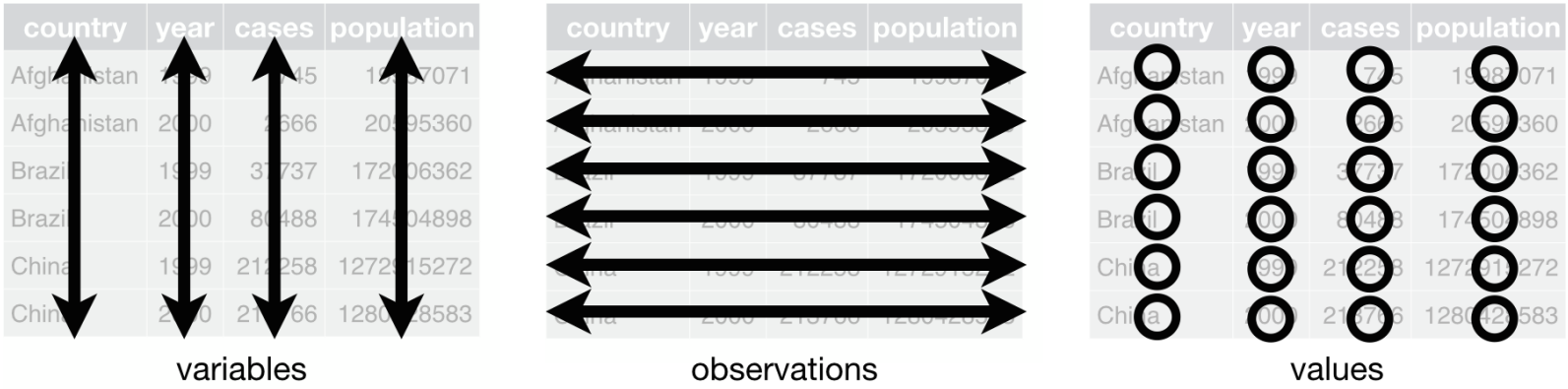
# The ‘tidyverse’

The tidyverse is set of R packages that operate ‘tidy data’

- i.e., that operate on data frames (or tibbles)

Tidy data is data where:

- Each variable must have its own column
- Each observation must have its own row
- Each value must have its own cell



# Messy data…

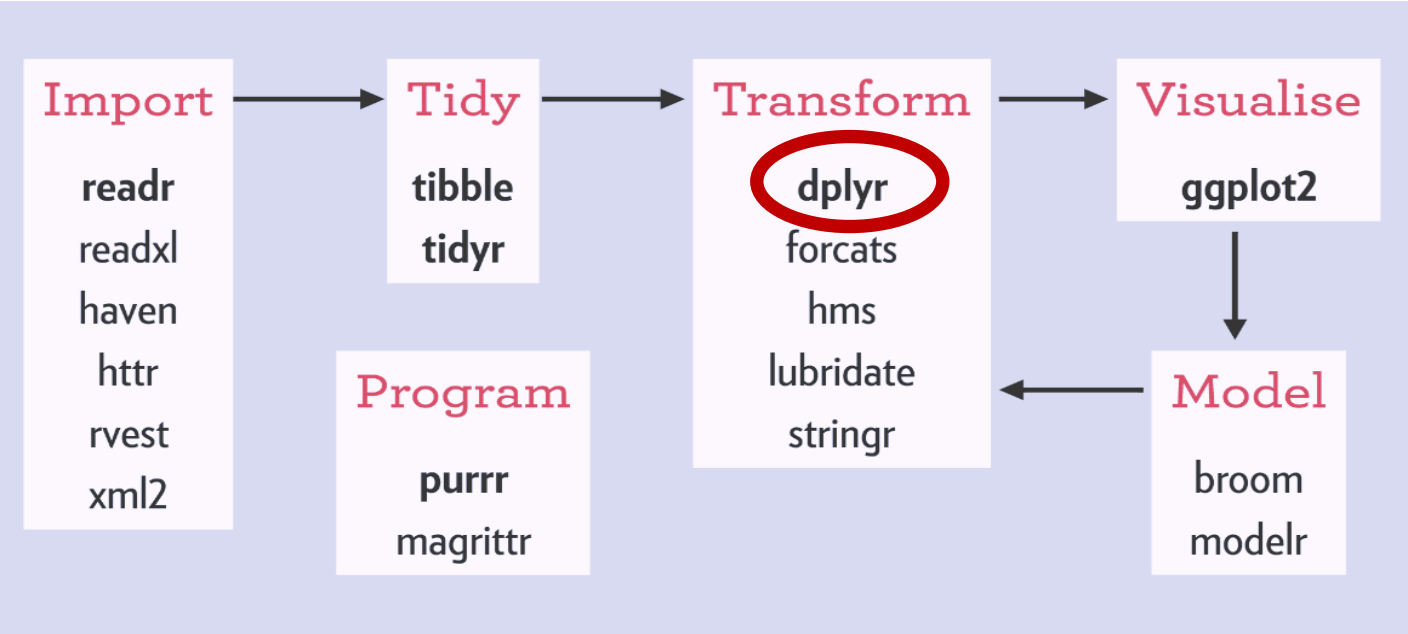
Messy data can be difficult to deal with

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Curve information - Curve quality data | | | | | | | | | | | | | | | |
| Name | Formula | Slope at | Intercept | ED-20 | ED-50 | ED-80 | Correlatic | Forced through origo | | | | | | | |
| Standard | Calc 1: C | standarc | standarc | 3792394 | 27752 | 0.2 | 0.5 | 0.8 | 1 | No | | | | | |
| | | | | | | | | | | | | | | | |
| Plate information | | | | | | | | | | | | | | | |
| Plate | Repeat | Barcode | Measurec | Chamber | Chamber | Humidity | Humidity | Ambient | Ambient | Formula | Measurement date | | | | |
| 1 | 1 | | N/A | N/A | N/A | N/A | N/A | N/A | N/A | Calc 1: C | standarc | standarc | 10.12.2013 10:23:33 | | |
| | | | | | | | | | | | | | | | |
| Background information | | | | | | | | | | | | | | | |
| Plate | Label | Result | Signal | Flashes/1 | Meastime | MeasInfo | | | | | | | | | |
| 1 | PicoGree | 0 | 110307 | 10 | 0 | De=1st Ex=Top Em=Top Wdw=N/A | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| Calculate | standarc | standards on each plate) where Label: PicoGreenFilterTop(1) channel 1 | | | | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | | | |
| A | -0.0011 | -0.0011 | -0.001 | -0.001 | -0.0011 | -0.0012 | -0.0011 | -0.0011 | -0.0012 | -0.0012 | 0.9973 | 1.0026 | | | |
| B | 0.0012 | 0.0014 | 0.0013 | 0.0012 | 0.0013 | 0.0012 | 0.0014 | 0.0003 | -0.0011 | -0.0011 | 0.0981 | 0.103 | | | |
| C | 0.0016 | 0.0013 | 0.0013 | 0.0011 | 0.0012 | 0.0015 | 0.0016 | -0.0004 | -0.0011 | -0.0011 | 0.0104 | 0.0095 | | | |
| D | 0.0019 | 0.0024 | 0.0018 | 0.0015 | -0.001 | -0.001 | -0.001 | -0.001 | -0.0011 | -0.0011 | 0.0008 | 0.0009 | | | |
| E | -0.001 | -0.0011 | -0.0011 | -0.0011 | -0.001 | -0.0012 | -0.0011 | -0.001 | -0.0009 | -0.0011 | -0.0001 | -0.0002 | | | |
| F | -0.001 | -0.0011 | -0.001 | -0.001 | -0.0012 | -0.0011 | -0.0011 | -0.0009 | -0.001 | -0.001 | -0.0003 | -0.0002 | | | |
| G | -0.0011 | -0.0011 | -0.0011 | -0.001 | -0.001 | -0.0012 | -0.0011 | -0.001 | -0.001 | -0.0011 | -0.0002 | 0.0012 | | | |
| H | -0.0011 | -0.0012 | -0.0011 | -0.001 | -0.0011 | -0.0011 | -0.0012 | -0.0011 | -0.0011 | -0.001 | -0.0003 | -0.0003 | | | |
| | | | | | | | | | | | | | | | |

# The ‘tidyverse’

The tidyverse is a set of packages share a common design philosophy

- Most written by Hadley Wickham



# dplyr: A grammar for data wrangling

**Grammar**: a set of components that can be combined to achieve a goal

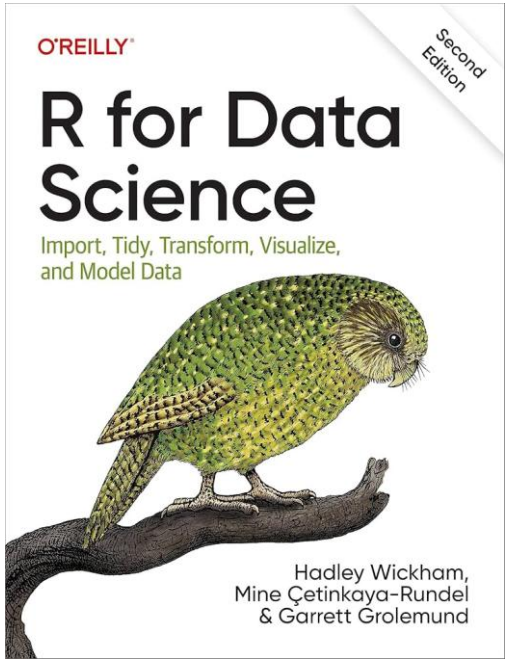
**dplyr** is a package that has a set of verbs that are useful for transformations data:

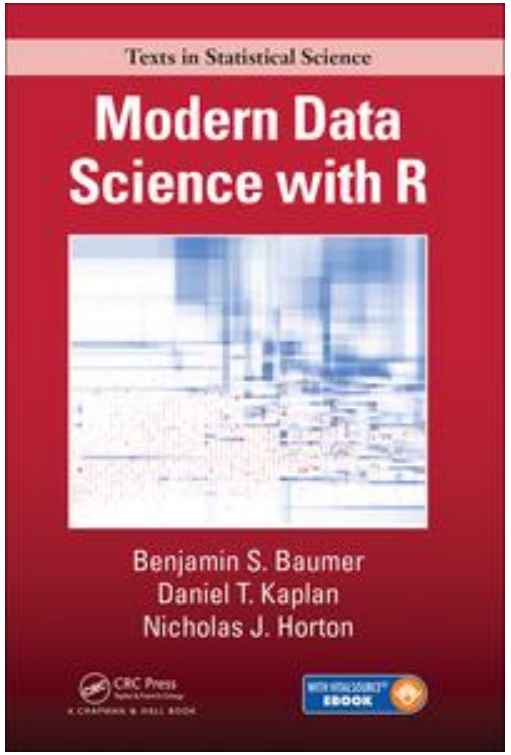
1. filter()
2. select()
3. mutate()
4. arrange()
5. group_by()
6. summarize()

All these function **take a data frame** and other arguments and **return a data frame**

```
> library(dplyr)   # load the dplyr package
```

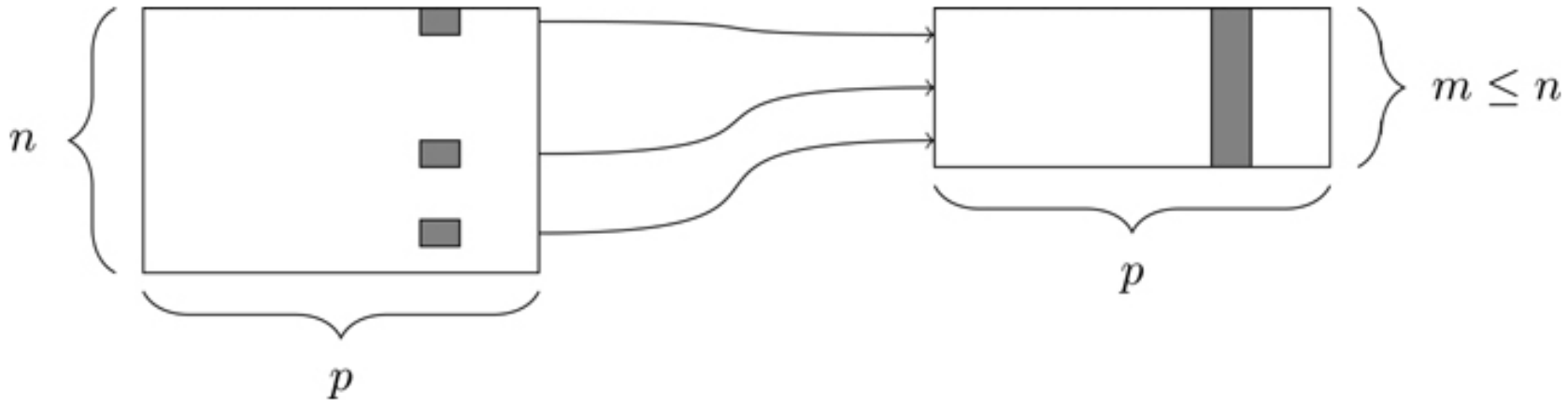
# Quick overview of the dplyr functions

# 1. filter()

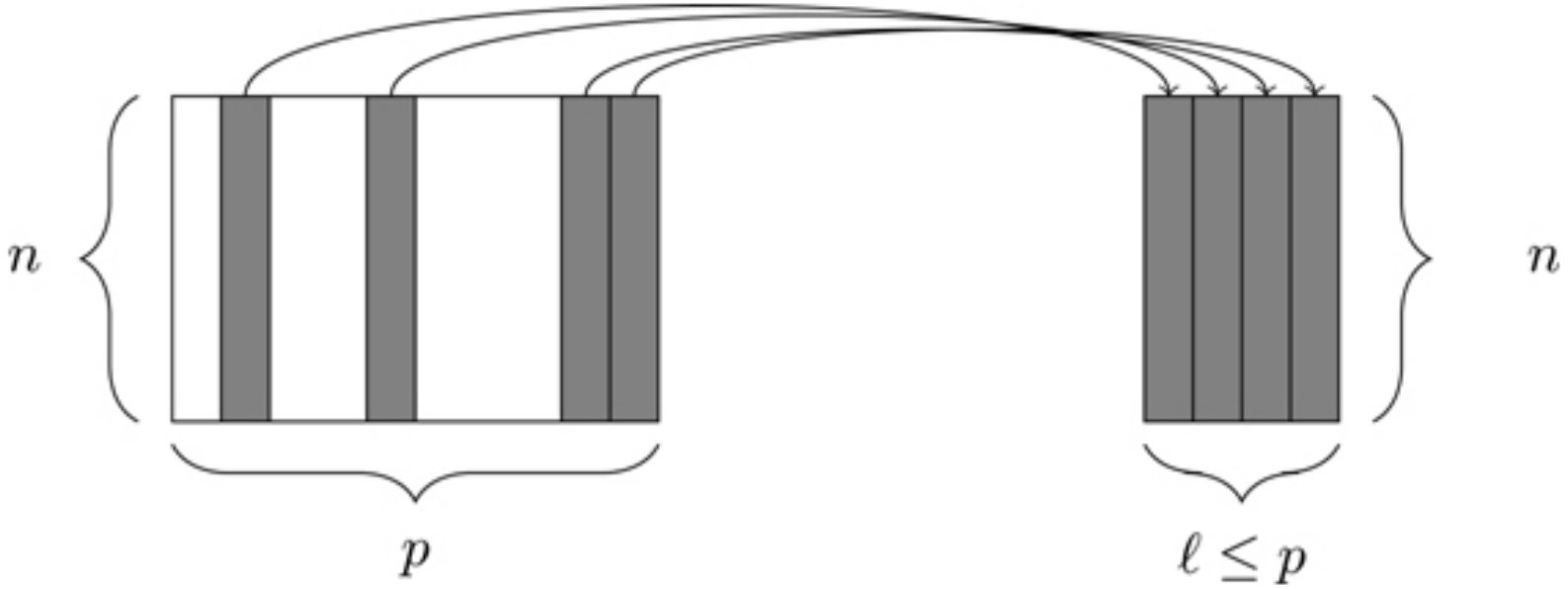
The filter() function allows you to select a subset of rows in data frame




filter(profiles, height == 77)

# 2. select()

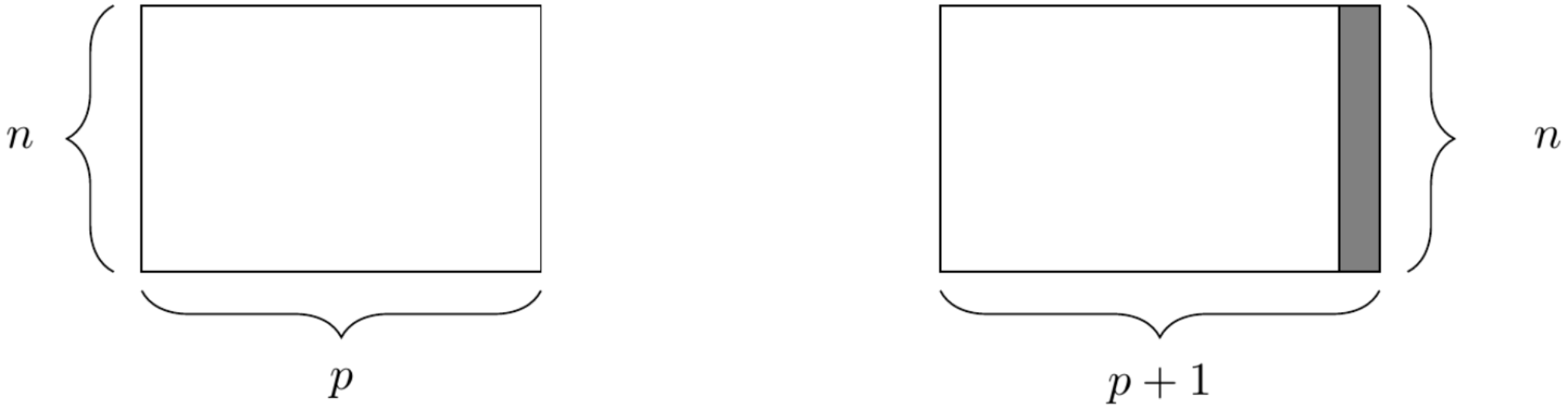
The select() function allows you to select a subset of columns




select(profiles, age, height)

# 3. mutate()

The mutate() function allows you to create new columns that are functions of existing columns

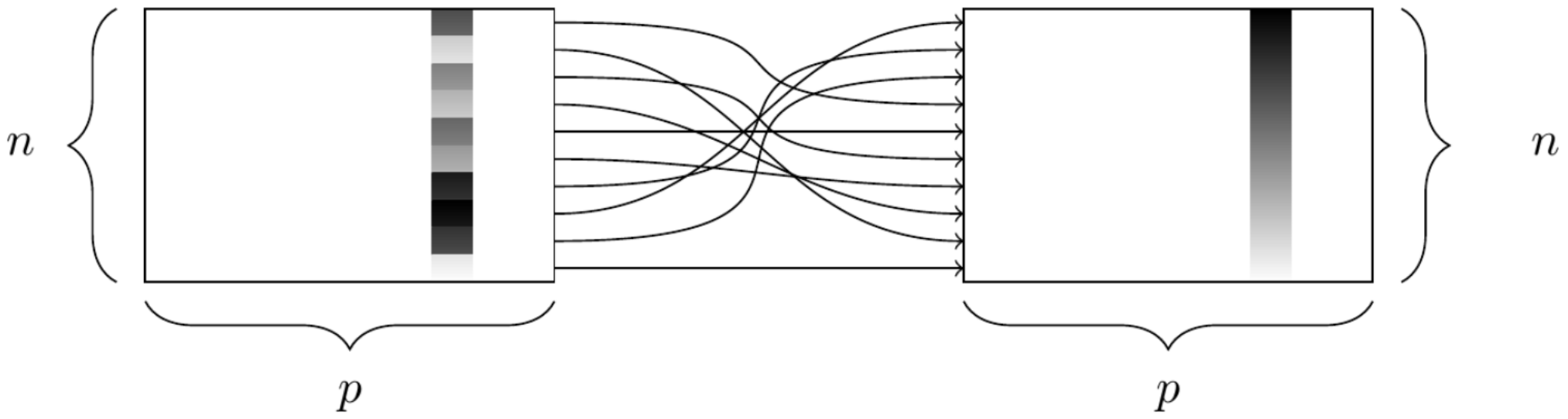



mutate(profiles, height_feet = height/12)

# 4. arrange()

The arrange() function arranges the rows based values in a column

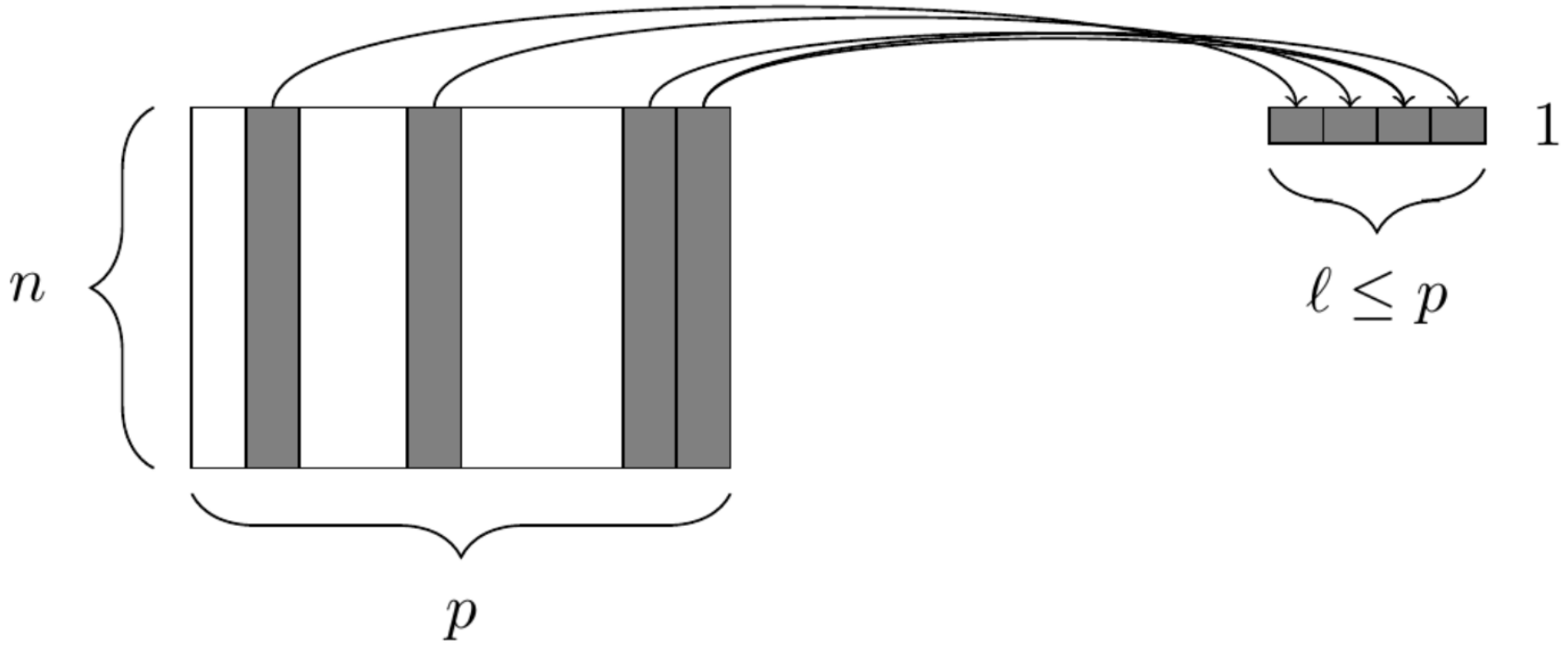
- arrange(desc()) arranges from largest to smallest




arrange(profiles, desc(height))

# 5. summarize()

The summarize() function reduces values in many rows into single values

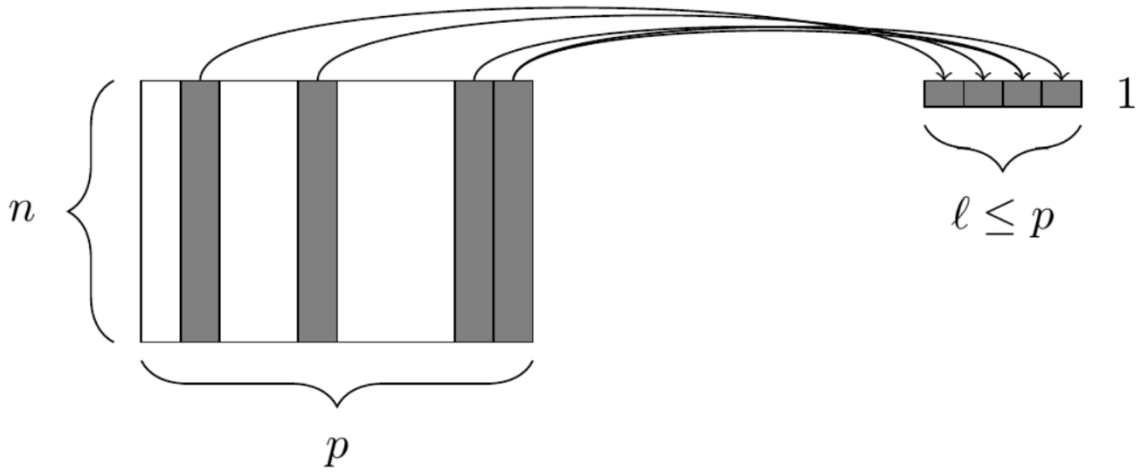



summarize(mean_age = mean(age))

# 6. The group_by() function

The group_by() function groups variables for future operations

- It works in conjunction with summarize() and mutate()
- It is used to do split, apply, combine



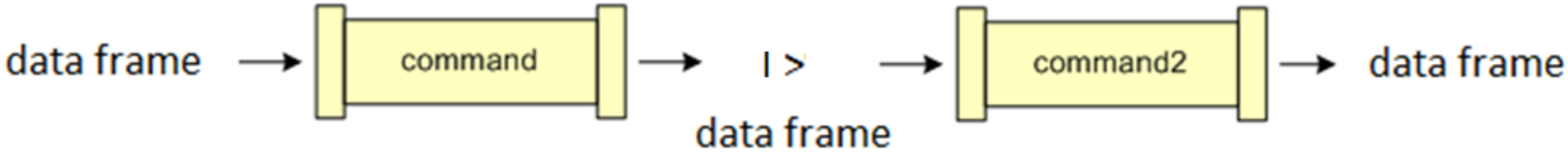

Input Data

| x | y |
|---|---|
| a | 2 |
| a | 4 |
| b | 0 |
| b | 5 |
| c | 5 |
| c | 10 |

group_by(profiles, sex)

# The pipe operator

The pipe operator |> allows us to chain commands together




```
profiles|>
        group_by(sex) |>
        summarize(mean_age = mean(age))
```


Let’s try it out!

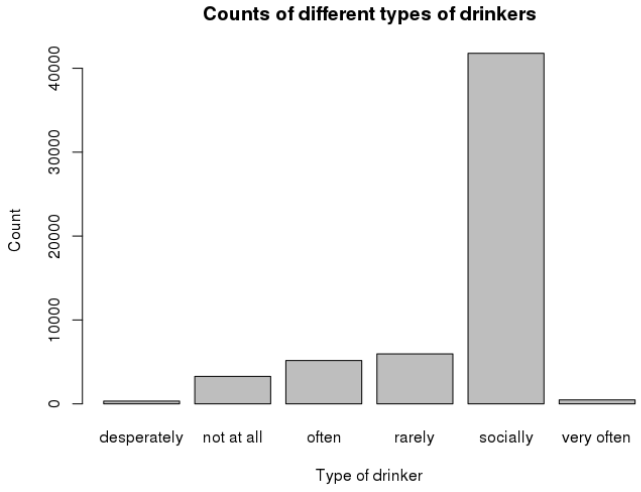
# A grammar of graphics and ggplot

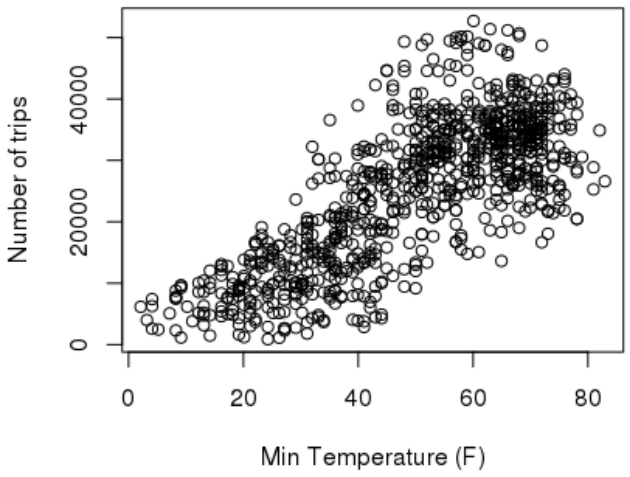
How have we plotted a single categorical variable?

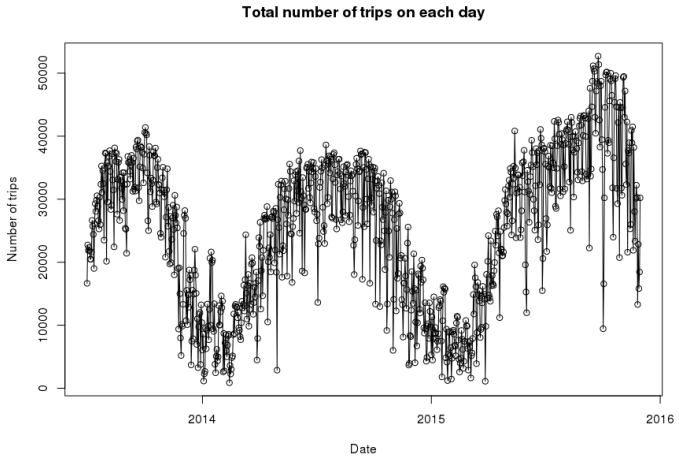
How have we plotted a single quantitative variable?

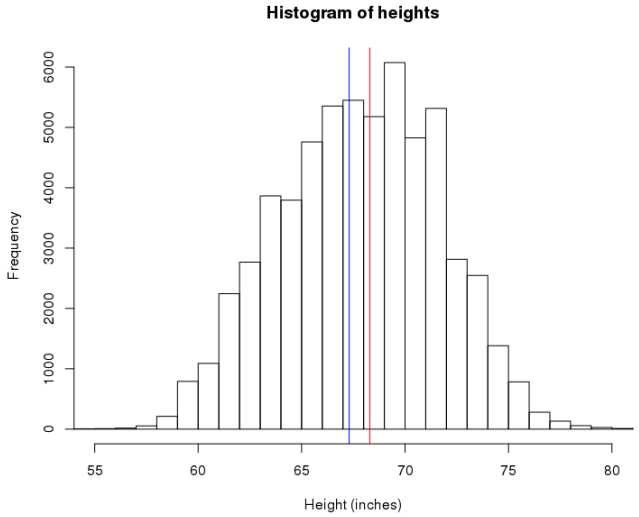
How have we plotted a two quantitative variables?

# What are some similarities between these graphs?



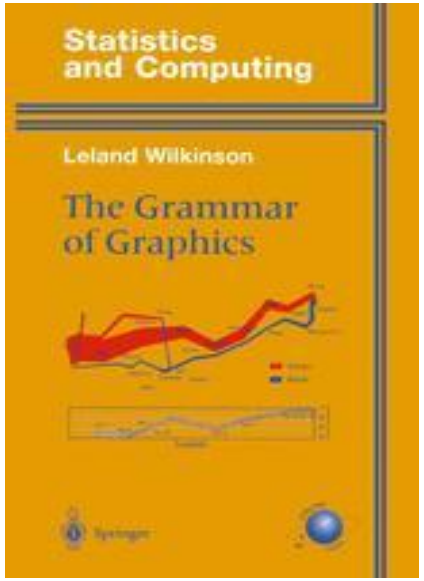



# The grammar of graphics

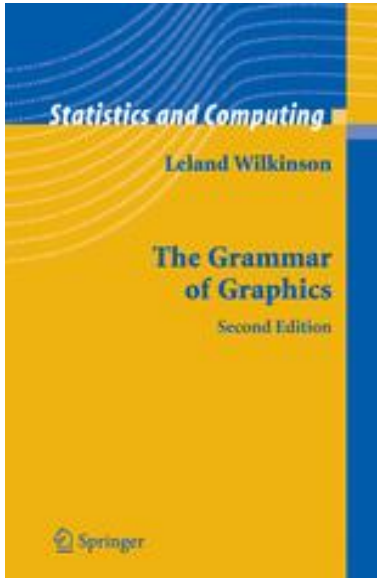
Leland Wilkinson noticed similarities between many graphs and tried to generate a ‘grammar’ that could be used to express a graph

- i.e., a list elements that can be combined together to create a graph

First edition




Second edition



# Graphs are composed of…

**A Frame**: Coordinate system on which data is placed

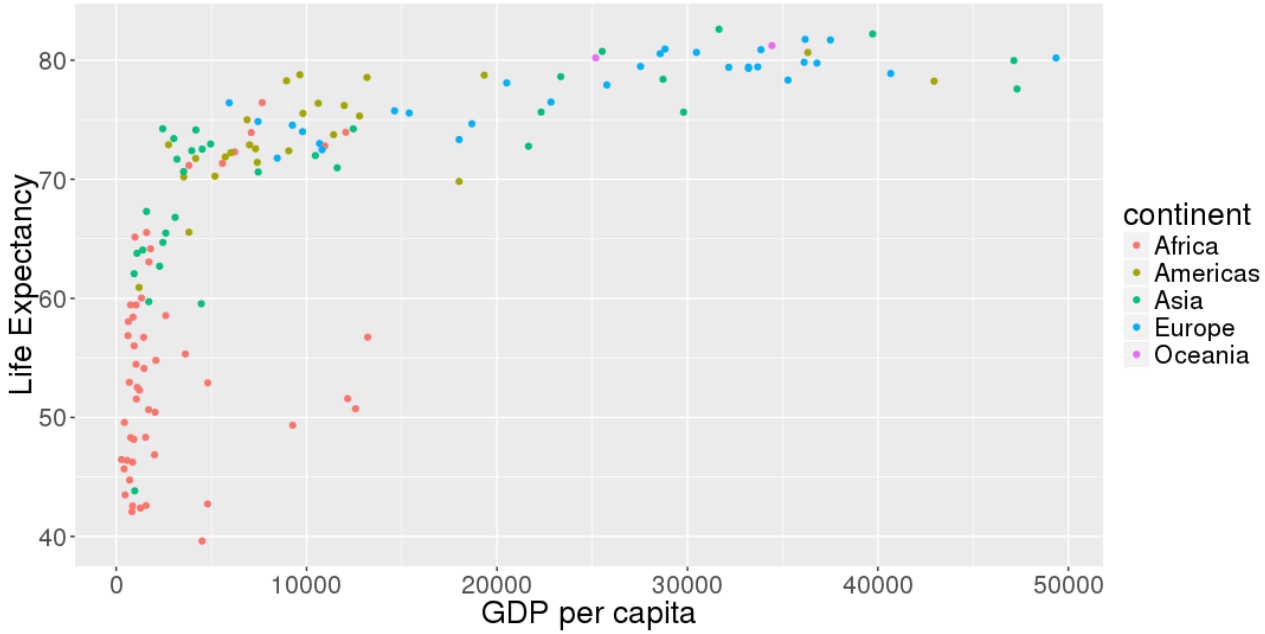
- E.g., Cartesian coordinate system, polar coordinates, etc.

**Glyphs**: basic graphic unit representing cases or statistics

- Contains visual properties (aesthetics) such as:  position, shape, color, size, etc.
- Need to specify how properties of the data are **mapped** onto these aesthetics

**Scales and guides**: shows how to interpret axes and other properties of the glyphs

- i.e., tells us how the data values are mapped into glyph properties



# Plots can also contain…

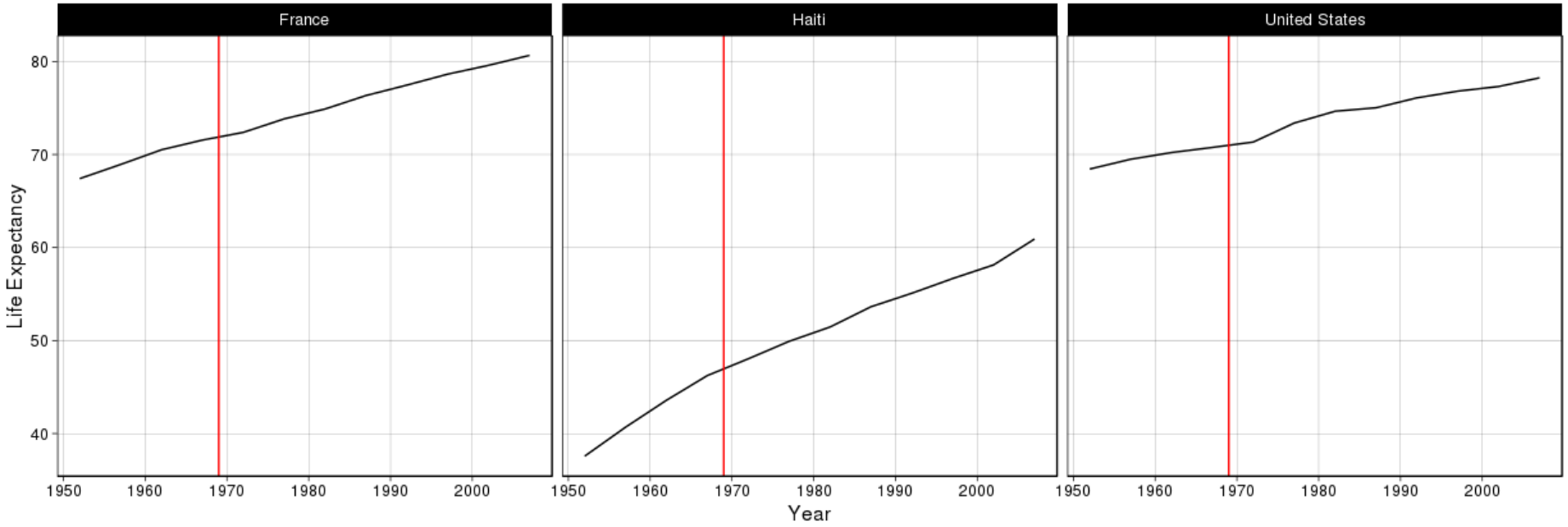
**Facets**: allows for multiple side-by-side graphs based on a categorical variable

- Makes it easier to compare different conditions

**Layers:** allows for more than one types of data to be mapped onto the same figure

**Theme**: contains finer points of display

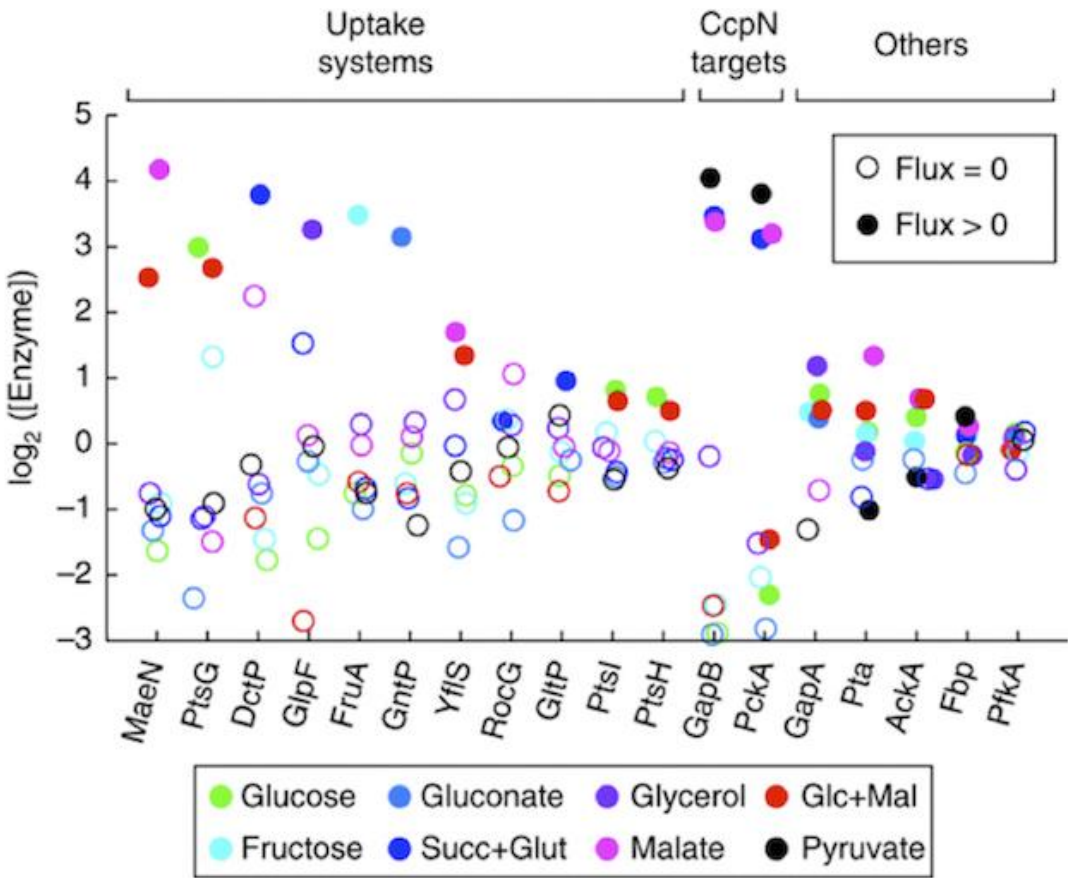
- E.g., font size, background color, etc.



# Aesthetic mappings

Data Frame

| Log_Enzyme | Gene | Target | Flux | Molecule |
|---|---|---|---|---|
| 4.5 | MaeN | Uptake | Positive | Glucose |
| -0.2 | PtsG | Uptake | Zero | Glucose |
| 3.8 | PckA | CcpN | Positive | Pyruvate |
| -0.1 | MaeN | Uptake | Zero | Gluconate |
| 3.2 | GntP | Uptake | Positive | Glycerol |



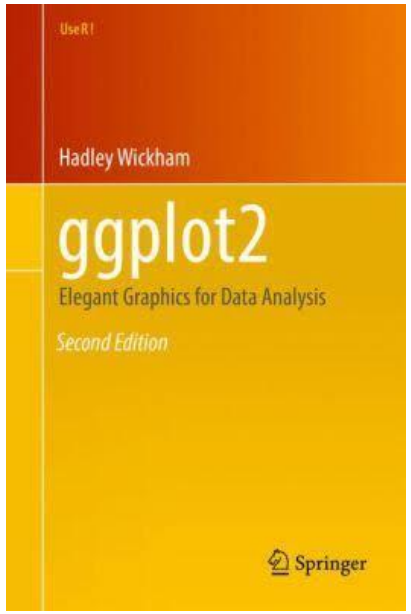

**Q: What are the mappings between each variable and visual attribute?**

# ggplot

**ggplot2** is an R package that implements the grammar of graphics

- It builds up graphics by starting with a frame, adding glyphs, etc.

```
# load the ggplot2 library
> library('ggplot2')
```




Get the book on GitHub

# Example data: mtcars

ROAD TEST By Jim Brokaw

## THE LUXURY CARS

**Imperial Palace**
**Fortress Fleetwood**
**Castle Continental**

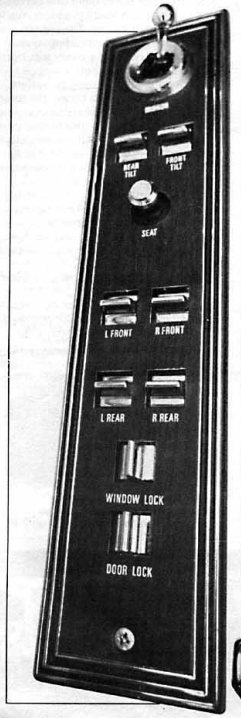
Crippled by the fuel flap and sniggered at lewdly by those smug Mercedes owners, today it seems that these great mastadons are dismissed as symbols of an ancient aristocracy whose strata was marked by expanse of wheelbase, and the heavenly quantity of gross cubic mass able to be shouldered by four beleagured tires. As the sole surviving heirs of princely Packards, dynosaurean Duesenbergs and the Brobdingnagian Bugattis, these marvels of grand proportion should be headed for the Smithsonian Institute by way of the mucky La Brea tar pits.

But are they?

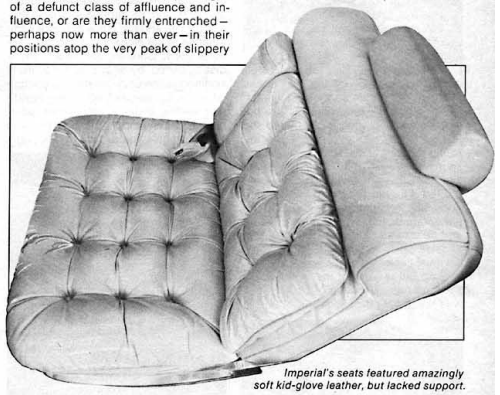
Are these behemoths simply vestiges of a defunct class of affluence and influence, or are they firmly entrenched – perhaps now more than ever – in their positions atop the very peak of slippery Mount Status, whose crags we scale daily, whether we wish to acknowledge that fact or not?

Ah, but welcome to the current state of American affairs: beef prices manipulated by withholding the animals from market; endless rounds of strikes by unions whose members are engaged in serving the public; gasoline supplies that rise and fall in mysterious coincidence with rising prices; dry rot, manipulation and partisan hatcheteering in government. They leave us little to believe in, and less to trust.

The very qualities that appear to condemn these sail-less luxury liners will very likely ensure their perpetuation. In days past, the block-long Cadillac and shiny Lincoln with paint jobs three feet deep flaunted a socio-economic position we could never hope to achieve.

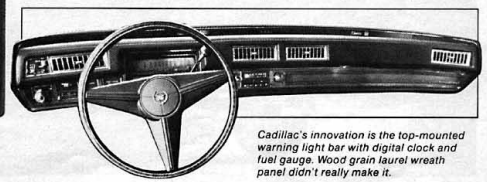



*Lincoln's placement of seat controls on arm rest panel is less desireable than lower side-of seat location of Cad and Imperial.*



*Imperial's seats featured amazingly soft kid-glove leather, but lacked support.*

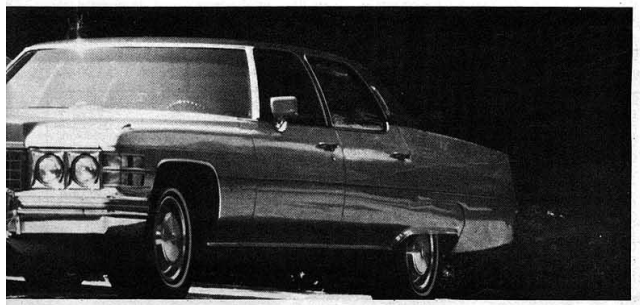
They did, however, constantly remind us that such positions, such wealth, such power, did, in fact, exist. The gliding specter of the shiny Imperial eagle stirred within a few heretical souls the bold idea that if such positions of power and wealth existed, there must be some means of attainment. More than a few of the haughty, distant drivers of the velvet tanks clawed their way up from the very pavement they now whisper over to



*Cadillac's innovation is the top-mounted warning light bar with digital clock and fuel gauge. Wood grain laurel wreath panel didn't really make it.*



| PERFORMANCE | CADILLAC | LINCOLN | IMPERIAL |
|---|---|---|---|
| **Acceleration** | | | |
| **0-30 mph** | 4.30 | 3.97 | 4.2 |
| **0-50 mph** | 8.49 | 8.00 | 9.15 |
| **0-60 mph** | 12.00 | 9.50 | 12.1 |
| **Standing Start 1/4-mile Mph** | 77.05 | 77.65 | 80.28 |
| **Elapsed time** | 17.98 | 17.82 | 17.42 |
| **Passing speeds** | | | |
| **40-60 mph** | 6.58 | 5.9 | 7.1 |
| **50-70 mph** | 7.00 | 6.8 | 6.8 |
| **Stopping distance** | | | |
| **From 30 mph** | 32'1" | 31'4" | 27'5" |
| **From 60 mph** | 182'7" | 153'10" | 129'3" |
| **Gas mileage range** | 10.43 | 10.42 | 14.7 |
| **Width – in.** | 79.8 | 80.0 | 79.7 |
| **Front Track – in.** | 63.5 | 64.3 | 64 |
| **Rear Track – in.** | 63.3 | 64.3 | 63.7 |
| **Wheelbase – in** | 133.0 | 127.0 | 124.0 |
| **Overall length – in.** | 233.7 | 232.6 | 231.1 |
| **Height – in.** | 55.6 | 55.4 | 54.7 |
| **Curb Weight – lbs.** | 5,250 | 5,425 | 5,345 |
| **Fuel Capacity – gals.** | 27 | 22.5 | 25 |
| **Oil Capacity – qts.** | 4 (1) | 4 (1) | 4 (1) |
| **Storage Capacity – cu. ft.** | 19.27 | 20.9 | 20+ |
| **Base Price** | $9,312 | $7,637 | $7,062 |
| **Price as tested** | $11,435 | $9,452 | $8,737 |
| **Engine:** | OHV V-8 | OHV V-8 | OHV V-8 |
| **Bore & Stroke – ins.** | 4.3x4.06 | 4.36x3.85 | 4.32x3.75 |
| **Displacement – cu. in.** | 472 | 460 | 440 |
| **HP @ RPM** | 205 @ 3600 | 215 @ 4000 | 230 @ 4000 |
| **Torque: lbs.-ft. @ rpm** | 365 @ 2000 | 350 @ 2600 | 350 @ 3200 |
| **Compression Ratio** | 8.25:1 | NA | 8.2:1 |
| **Carburetion** | 4V | 4V | 4V |
| **Transmission** | Auto. Turbo Hydra-Matic | Auto. Select Shift | Auto. Torqueflite |
| **Final Drive Ratio** | 2.93 | 3.00 | 3.23 (?) |
| **Steering Type** | Recirculating Ball & Nut Power | Recirculating Ball & Nut With Integral Power Unit | Recirculating Ball Power |
| **Steering Ratio** | 17.8-9.0 | 21.6 To 1 | 18.9:1 |
| **Turning Diameter (curb-to-curb-ft.)** | (Wall To Wall) 24.54' | 46.7' | 44.69' |
| **Wheel Turns (lock-to-lock)** | 2.83 | 3.99 | 3.5 |
| **Tire Size** | LR78X15 Steel Belted Radials | LR78X15 Steel Belted Radials | LR78X15 Steel Belted Radial Ply |
| **Brakes** | Power Disc/Drum | Power Disc/Drum | Power Disc/Disc |
| **Front Suspension** | Coils/Shocks Front Diagonal Tie Struts Stabilizer | Coils/Shocks Axial Strut Stabilizer | Torsion Bar Shocks Stabilizer |
| **Rear Suspension** | 4 Link, Coils/ Shocks | Three Link, Rubber Cushioned Pivots Coils/Shocks | Leaf Springs Shocks |
| **Body/Frame Construction** | Perimeter Frame | Body On Perimeter Frame | Unitized Construction |



# mtcars data frame

How can you determine what variables are in a data frame?

```
> View(mtcars)      # only works in Rstudio, not in Markdown
> glimpse(mtcars)
> ? mtcars      # this data frame as a code book
```

| | | |
|---|---|---|
| [, 1] | mpg | Miles/(US) gallon |
| [, 2] | cyl | Number of cylinders |
| [, 4] | hp | Gross horsepower |
| [, 6] | wt | Weight (1000 lbs) |
| [, 9] | am | Transmission (0 = automatic, 1 = manual) |

# Do cars that weigh more use more fuel?

**Question**: do cars that weigh more use more fuel?

What variables in the mtcars data frame are of interest?

- mpg
- wt

We can create a scatter plot using base graphics…

```
> plot(mtcars$wt, mtcars$mpg)
```

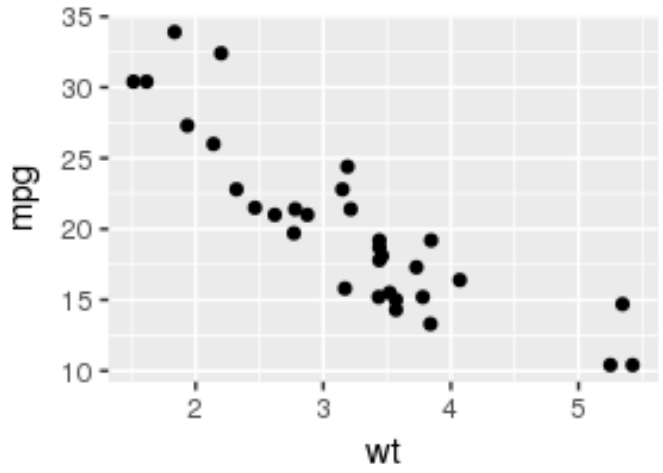
# Creating a scatter plot in ggplot

Data frame to be used

Aesthetic mapping

```
> ggplot(data = mtcars, mapping = aes(x = wt, y = mpg)) +
    geom_point()
```

Adds a layer with glyphs




| | wt | cyl | hp | mpg | disp |
|---|---|---|---|---|---|
| **Mazda RX4** | 2.620 | 6 | 110 | 21.0 | 160.0 |
| **Mazda RX4 Wag** | 2.875 | 6 | 110 | 21.0 | 160.0 |
| **Datsun 710** | 2.320 | 4 | 93 | 22.8 | 108.0 |
| **Hornet 4 Drive** | 3.215 | 6 | 110 | 21.4 | 258.0 |
| **Hornet Sportabout** | 3.440 | 8 | 175 | 18.7 | 360.0 |

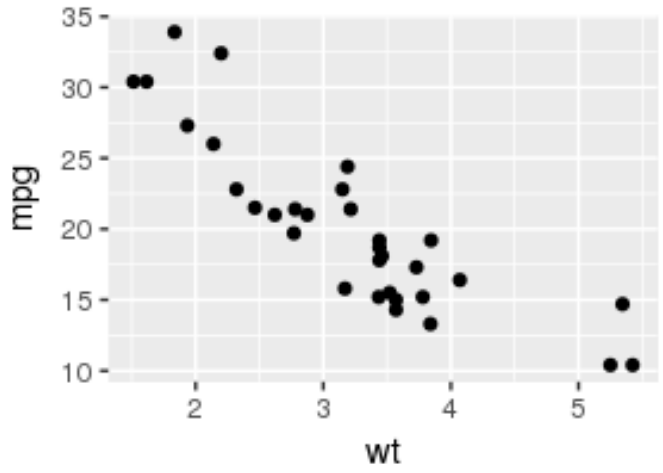
# Creating a scatter plot in ggplot

Data frame to be used

Aesthetic mapping

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
    geom_point()
```

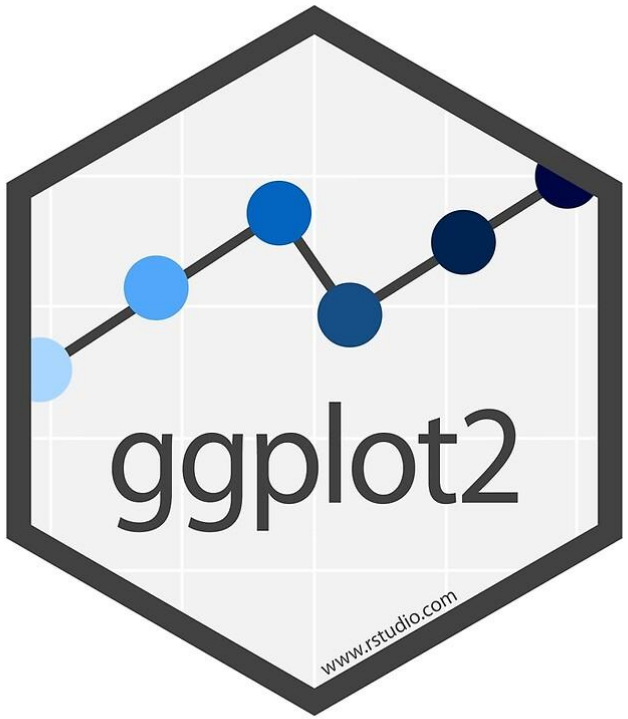
Adds a layer with glyphs




| | wt | cyl | hp | mpg | disp |
|---|---|---|---|---|---|
| **Mazda RX4** | 2.620 | 6 | 110 | 21.0 | 160.0 |
| **Mazda RX4 Wag** | 2.875 | 6 | 110 | 21.0 | 160.0 |
| **Datsun 710** | 2.320 | 4 | 93 | 22.8 | 108.0 |
| **Hornet 4 Drive** | 3.215 | 6 | 110 | 21.4 | 258.0 |
| **Hornet Sportabout** | 3.440 | 8 | 175 | 18.7 | 360.0 |

A lot more that ggplot can do!

- More aesthetic mapping
- Multiple glyphs/layers
- Axis labels
- Facets
- Visual themes
- Different coordinate systems
- Etc.




The R Graph Gallery

# Let’s try the rest in R!

# Adding labels to plots

We can add labels to the plots using the lab() functions

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
        geom_point()  +
        labs(x = "Weight",
              y = "Miles per Gallon",
              title = "MPG as a function of weight",
              subtitle = "ggplot is cool")
```

# More aesthetic mappings

Let's look at the relationship between weight, miles per gallon and transmission type on the same graph by plotting… (?)

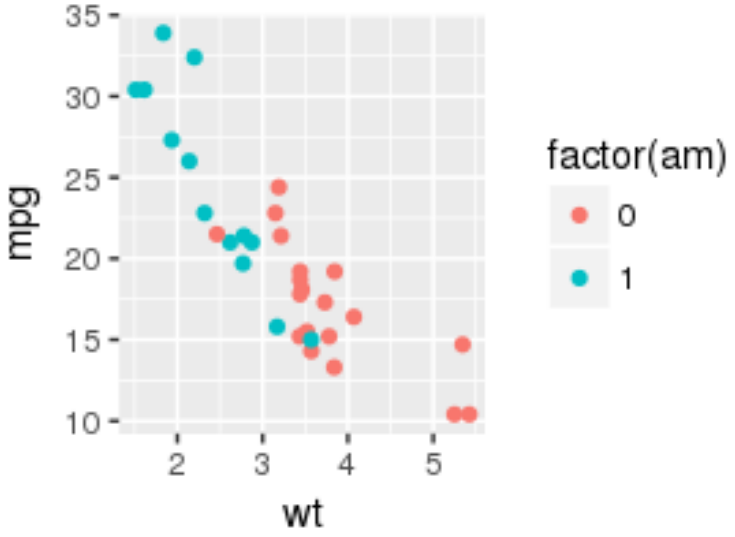
```
> ggplot(mtcars, aes(x = wt, y = mpg, col = am)) +
        geom_point()

# It is better if we make am a categorical variable
> ggplot(mtcars, aes(x = wt, y = mpg, col = factor(am))) +  geom_point()
```

Notice the guides!!!

Try mapping am on to shape using:

1. *shape = am*
2. size using: *size = am*

Which is better to use color or shape or size?



# Attributes vs. Aesthetics

Setting **aesthetics** map a variable to a glyph property

Setting **attributes** set a glyph property to a fixed value

```
# setting a aesthetic
> ggplot(mtcars) +
        geom_point(aes(x = wt, y = mpg, col = factor(am)))

# setting an attribute
> ggplot(mtcars) +
        geom_point(aes(x = wt, y = mpg), col = "red")
```

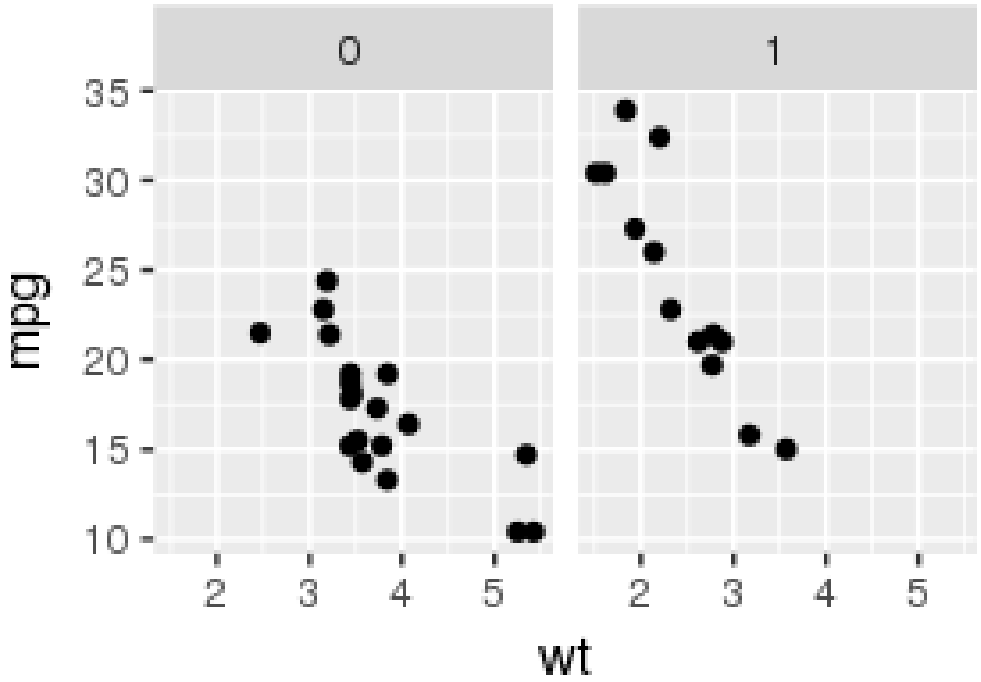
Outside the aesthetic mapping!

# Facets

Beyond comparing variables based on aesthetics you can compare categorical variables by splitting a plot into subplots (called facets) using facet_wrap

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
        geom_point() +
        facet_wrap(~am)
```

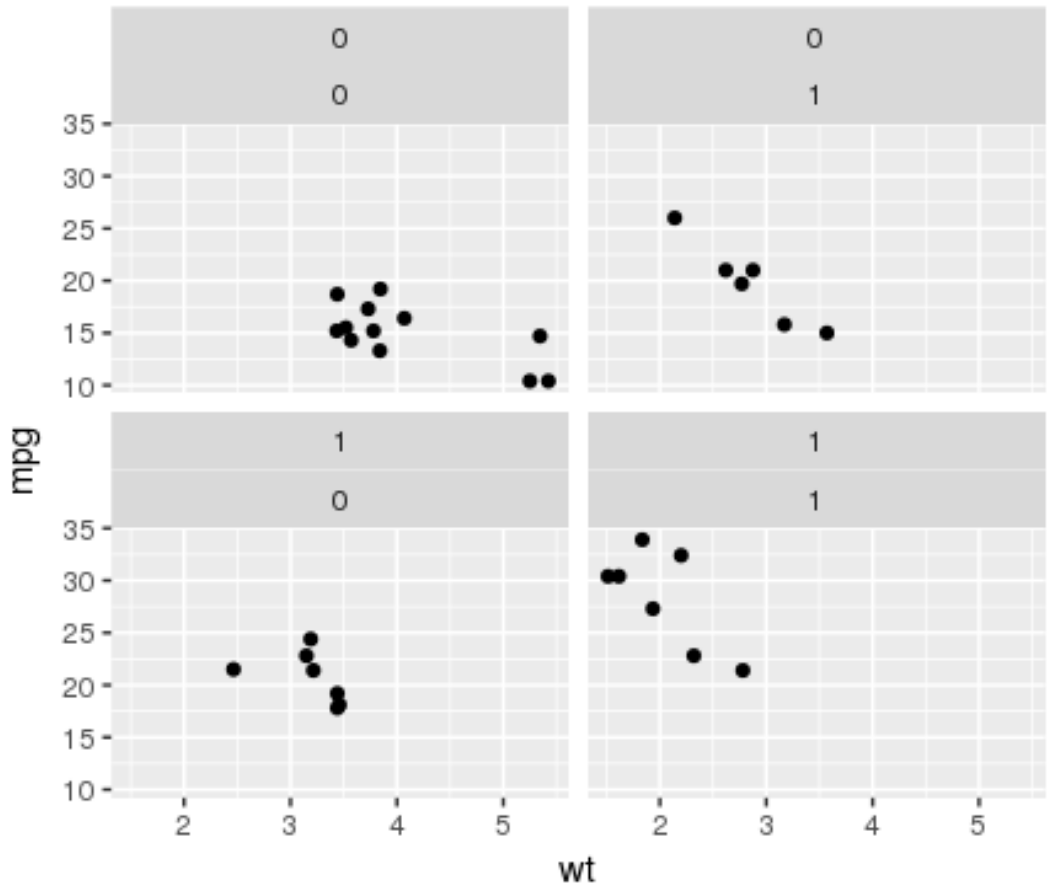
What do facets make it easy to see on this graph?



# Facets along two dimensions

One can also do facets in two dimensions

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
        geom_point() +
        facet_wrap(vs ~am)
```



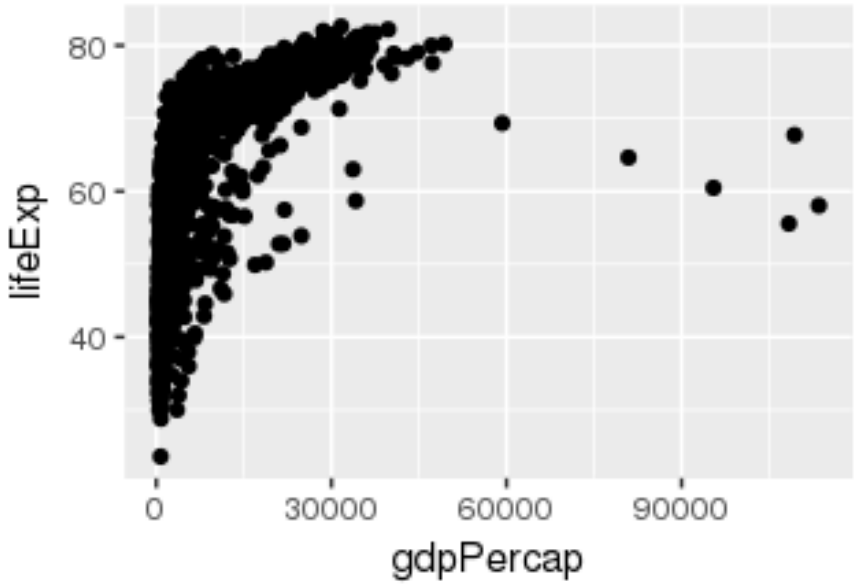
# Overplotting

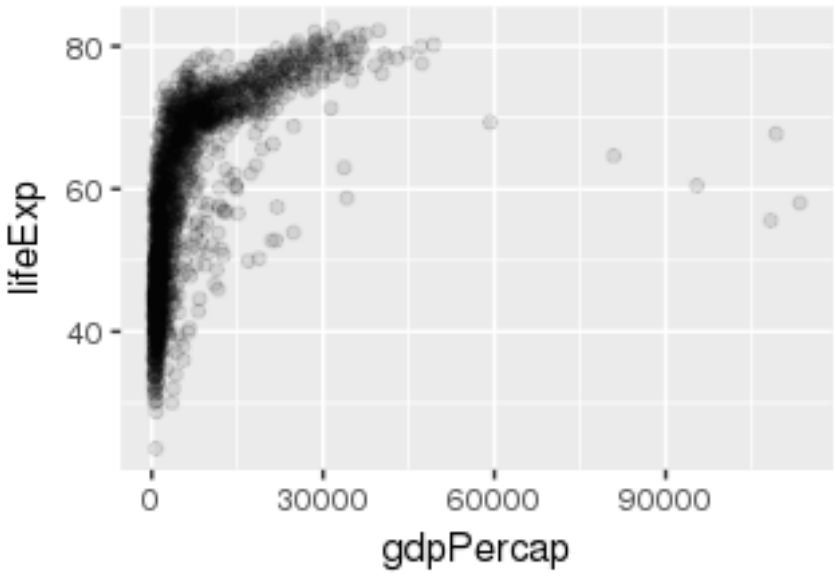
Sometimes points overlap making it hard to estimate the number of points at a particular range of values

We can control the transparency of points by changing their alpha values

```
# compare these two plots
> ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
        geom_point()

> ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
        geom_point(alpha = .1)
```

# Overplotting



# Scales

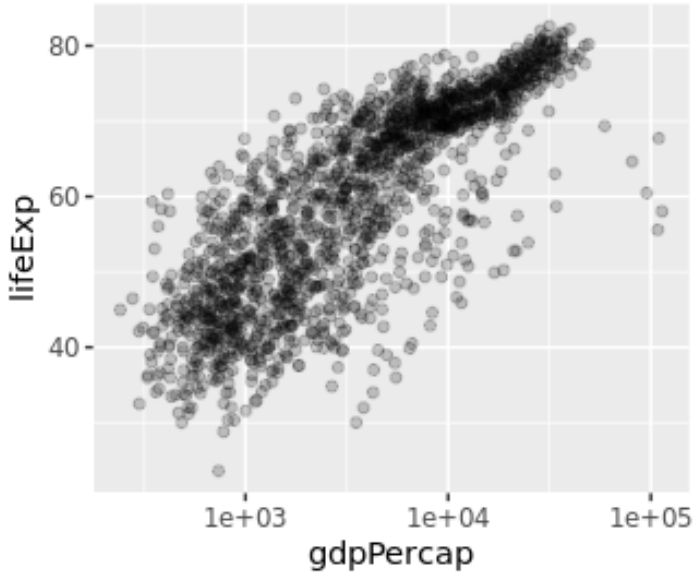
We can change the scale underlying each aesthetic visual feature

- We use functions that start with scale_ to do this

For example, we can change the x scale from linear to logarithmic using:

- scale_x_continuous(trans='log10')

```
> ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
                geom_point(alpha = .2) +
                scale_x_continuous(trans='log10')
```



# Scales

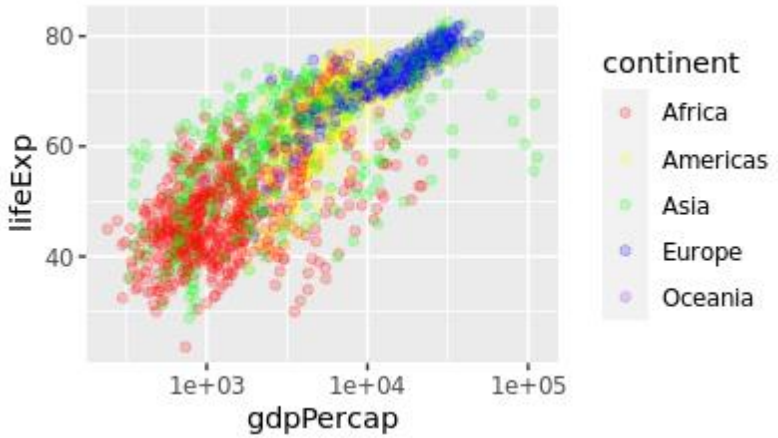
We can change the scale underlying each aesthetic visual feature

- We use functions that start with scale_ to do this

We can change the color scale using:

- scale_color_manual()

```
> ggplot(gapminder, aes(x = gdpPercap, y = lifeExp,
                        col =continent)) +
    geom_point(alpha = .2) +
    scale_color_manual(values = c("red", "yellow",
                                  "green", "blue", "purple"))
```



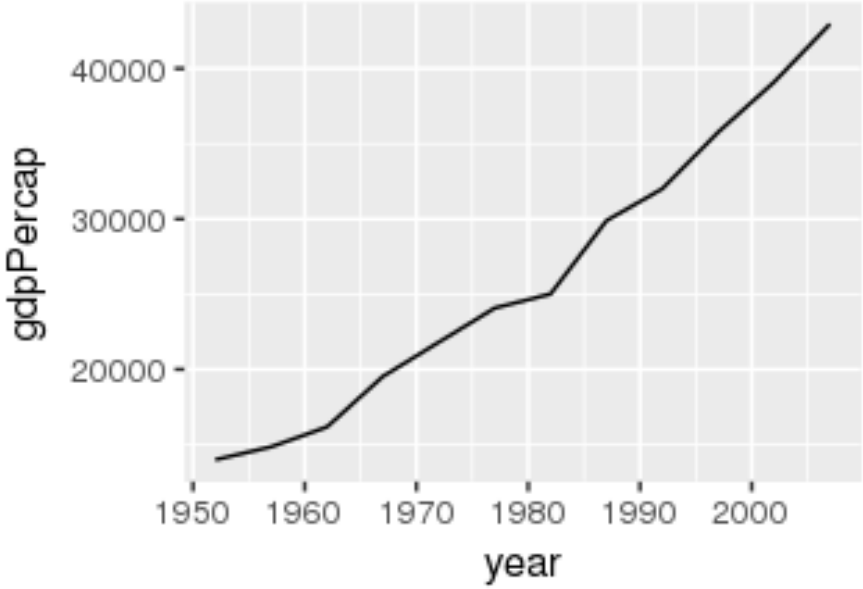
# Geometries: line plot

So far we've only created scatter plots, but we can use different geoms to create other types of plots

Create a plot that shows the GDP in the United States as a function of the year using the geom *geom_line()*

- Hint: filter the gapminder data first…

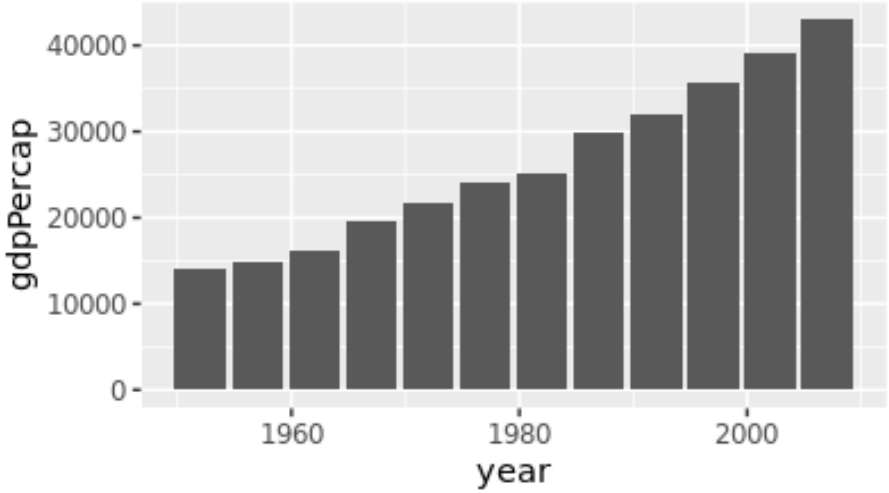
```
> gapminder |>
      filter(country == 'United States') |>
      ggplot(aes(x = year, y = gdpPercap)) +
      geom_line()
```



# Geometries: columns

Create a plot that shows the GDP in the United States as a function of the year as columns geom *geom_col()*

```
> gapminder |>
      filter(country == 'United States’) |>
      ggplot(aes(x = year, y = gdpPercap)) +
      geom_col()
```

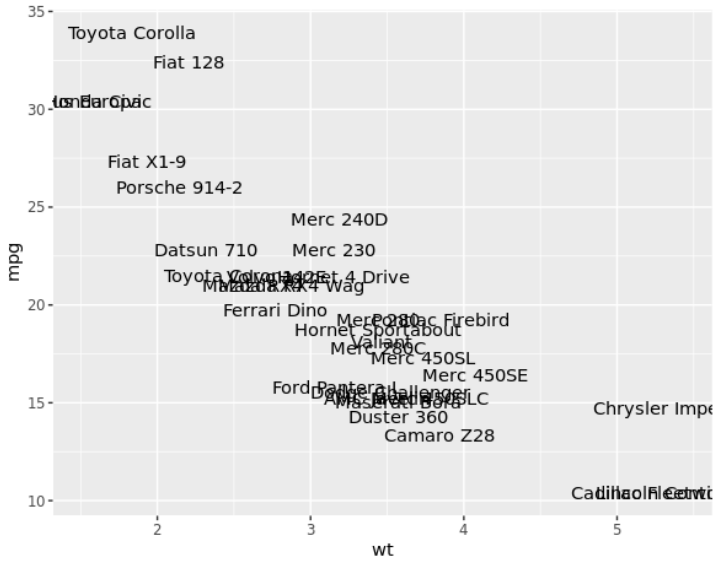


# Geometries: text

Create can also use text as a geom using *geom_text(aes(label = ))*

We will first add the row names as a column to our data frame using tibble::rownames_to_column()

```
> mtcars |>
      tibble::rownames_to_column() |>
      ggplot(aes(x = wt, y = mpg)) +
            geom_text(aes(label = rowname))
```



# Geometries: histograms

We can also make histograms using the geom_histogram() function.

Plot a histogram of the weights of cars

```
> ggplot(mtcars, aes(x = wt)) +
        geom_histogram()
```

Note the histogram geom only has an x aesthetic, and does not have a y aesthetic value.

# Geometries: boxplot

There are many other geom as well, including geom_boxplot()

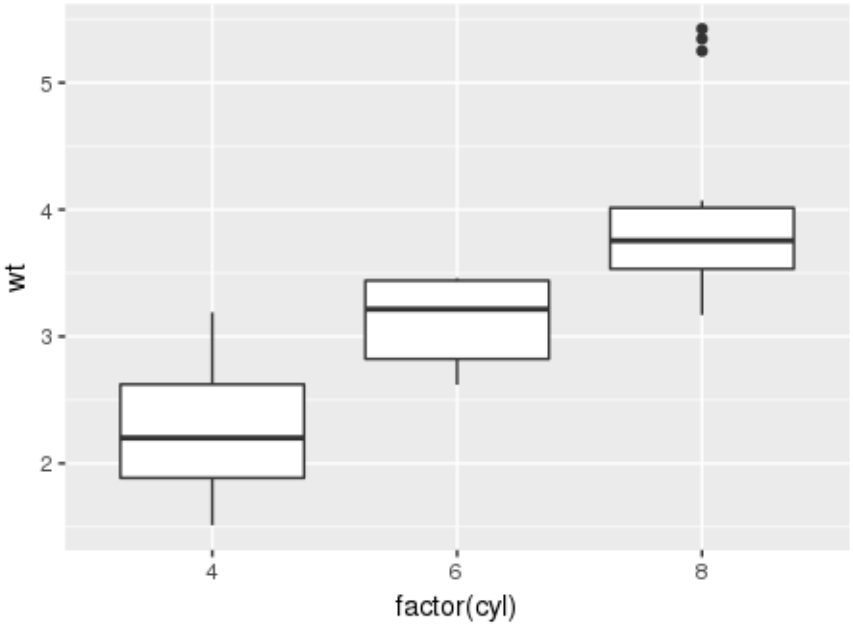
Plot a boxplot of the weights of cars

```
> ggplot(mtcars, aes(x = "", y = wt)) +
        geom_boxplot()
```

# Side-by-side boxplots

Often it is useful to compare boxplots across different groups

```
> ggplot(mtcars, aes(x = factor(cyl), y = wt)) +
        geom_boxplot()
```



# Violin and Joy plots

Violin and Joy plots are other ways to view distributions of data

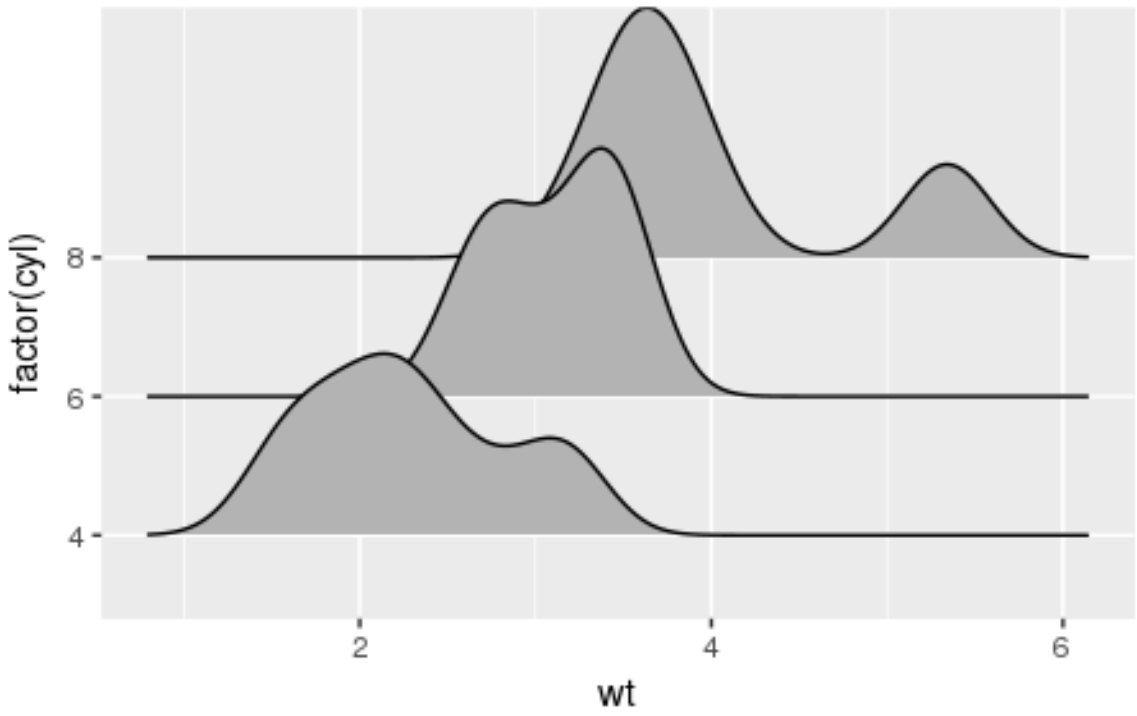
```
> ggplot(mtcars, aes(x = factor(cyl), y = wt)) +
        geom_violin()
```

Note x and y are reversed

```
> library("ggridges")
> ggplot(mtcars, aes(y = factor(cyl), x = wt)) +
        geom_density_ridges()
```

# Violin and Joy plots

Any ideas why they are called joy plots?



# Multiple layers

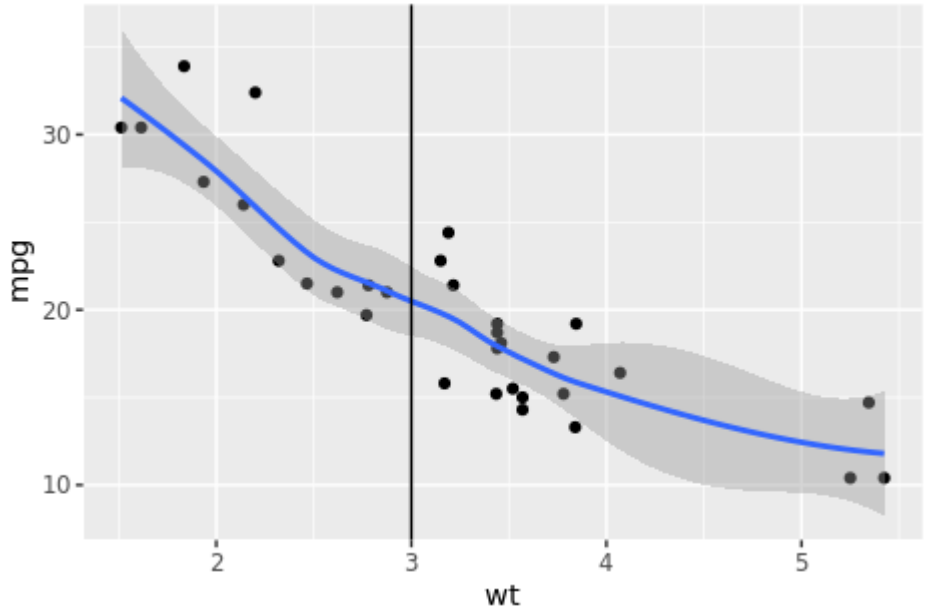
We can also have multiple geom layers on a single graph by using the + symbol

- E.g ggplot(…) + geom_type1() + geom_type2()

Create a scatter plot of miles per gallon as a function of weight and then add:

- a smoothed line using geom_smooth()
- a vertical line using geom_vline()

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
        geom_point() +
        geom_smooth() +
        geom_vline(xintercept = 3)
```



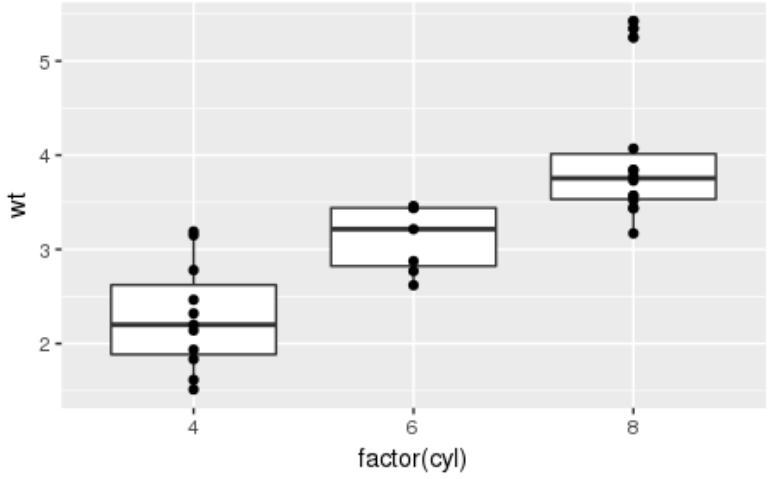
# Multiple layers

We can also have multiple geom layers on a single graph by using the + symbol

- E.g ggplot(…) + geom_type1() + geom_type2()

Recreate a boxplot of weight (wt) grouped by the factor of cylinders (cyl), and then add points using geom_point()

```
> ggplot(mtcars, aes(x = factor(cyl), y = wt)) +
        geom_boxplot() +
        geom_point()
```



# Themes

We can also use different types to change the appearance of our plot

Add theme_classic() to your plot

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
        geom_point()  +
        xlab("Weigth") +
        ylab("Miles per Gallon") +
        theme_classic()
```

Also see the theme_fivethirtyeight() from the ggthemes package

# Themes

We can also create a customized theme using theme()

```
> ggplot(mtcars, aes(x = wt, y = mpg)) +
        geom_point()  +
        theme_classic() +
        theme(
                axis.text.y = element_blank(),
                plot.background = element_rect(fill = "red")
            )
```