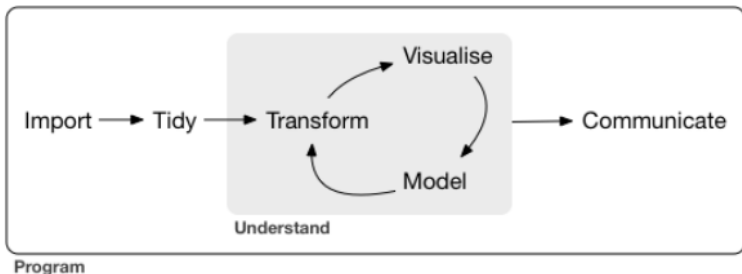


Stat 260, Lecture 2

Brad McNeney

Data Visualization

- ▶ Reading: Chapter 1 of the text.
- ▶ The key steps of a data analysis are illustrated in the following figure from the text:



- ▶ We will jump into the middle of this program and discuss data visualization, because it is often the most interesting.
- ▶ We will use the tidyverse package ggplot2 to visualize data.

Loading ggplot2

- ▶ Before you start, make sure you install the tidyverse collection of R packages. You can do this:
 - ▶ With the Tools -> Install Packages menu item in RStudio. Type gapminder, tidyverse into the text box and click Install.
 - ▶ By typing `install.packages("tidyverse")` into the R console.
- ▶ Install once, load every time with `library()`:

```
library(tidyverse)
```

Example: Car mileage

- ▶ Do cars with big engines use more fuel than cars with small engines?

```
data(mpg)
head(mpg)
```

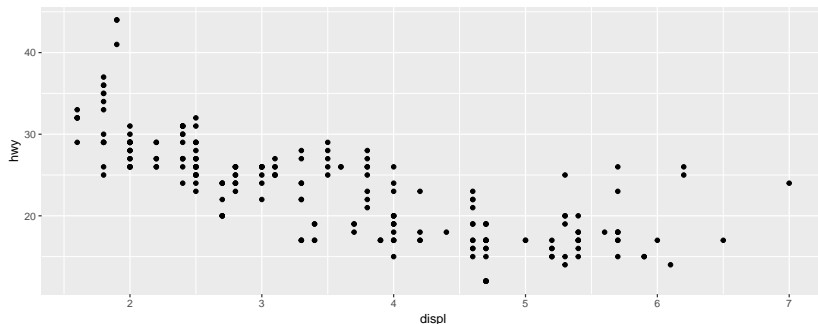
```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans  drv      cty   hwy fl      class
##   <chr>          <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 audi          a4      1.8  1999     4 auto(~ f      18    29 p      comp~
## 2 audi          a4      1.8  1999     4 manua~ f      21    29 p      comp~
## 3 audi          a4      2    2008     4 manua~ f      20    31 p      comp~
## 4 audi          a4      2    2008     4 auto(~ f      21    30 p      comp~
## 5 audi          a4      2.8  1999     6 auto(~ f      16    26 p      comp~
## 6 audi          a4      2.8  1999     6 manua~ f      18    26 p      comp~
```

- ▶ Type `View(mpg)` to view the dataset and `?mpg` for details on the variables.

First ggplot

- ▶ `displ` is the engine displacement in litres and `hwy` is the highway mileage in miles per gallon.
- ▶ We can plot `hwy` *versus* `displ` as follows:

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

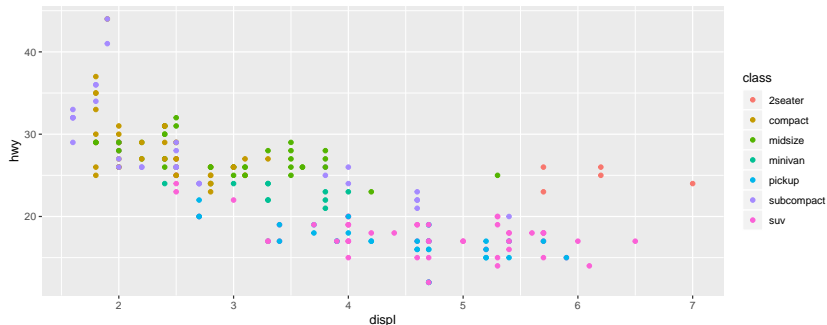


- ▶ Generally a negative linear trend, though there are some cars with large engines that get better-than-expected mileage.

Second ggplot

- ▶ We can plot *hwy* versus *displ* with colors to represent different kinds of cars:

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



- ▶ The 2seaters are sports cars, with large engines but light bodies.
- ▶ **Exercise:** Redo the above scatterplot but using the aesthetic `shape=class` to plot different shapes for different kinds of cars.

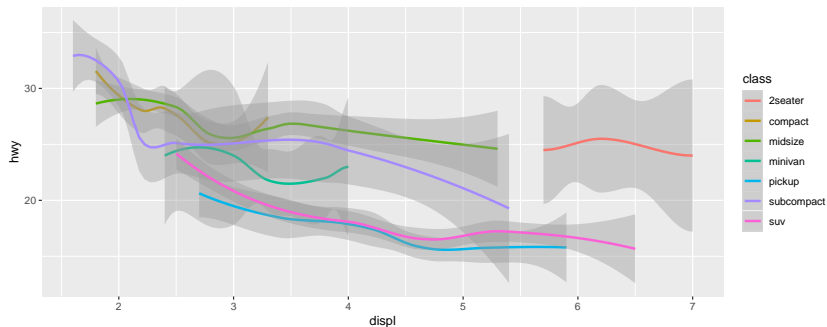
Components of our ggplot

- ▶ Our plot requires a dataset, objects to plot for each observation in the dataset, and a mapping of the variables in that dataset to features of the plot.
- ▶ In `ggplot(data = mpg)` we use the dataset `mpg`,
- ▶ we plot points for each observation in the dataset with `geom_point()`,
- ▶ and `mapping = aes(x=displ,y=hwy,color=class)` maps engine displacement to the x-axis, highway mileage to the y-axis and vehicle class to the color of each point.

Geometric objects

- ▶ The geometric objects to plot, or “geoms” are specified by the functions `geom_XXX()`.
- ▶ Example: scatterplot smooths for each class of car.

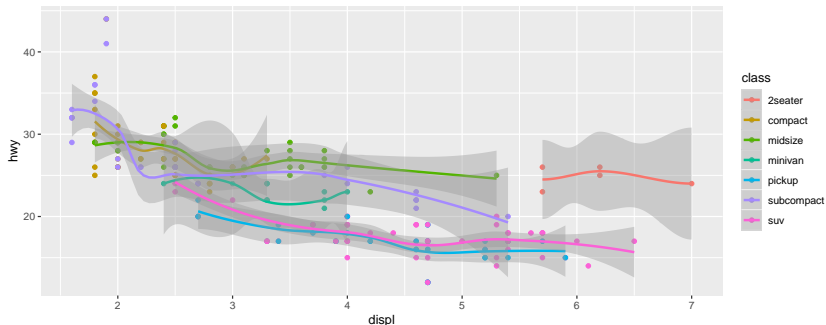
```
ggplot(data=mpg) + geom_smooth(mapping = aes(x=displ,y=hwy,color=class))
```



Multiple geoms

- ▶ Scatterplot smooths are more often plotted over top of points, as a summary of the trend.
- ▶ In the following we specify a default aesthetic mapping that is used by both the points and smooth.

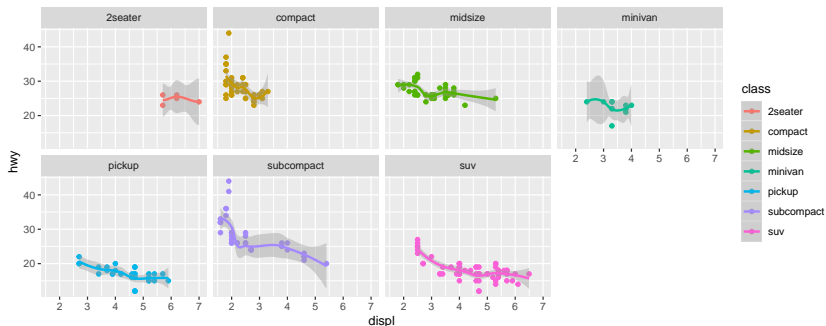
```
ggplot(data=mpg, mapping = aes(x=displ,y=hwy,color=class)) +  
  geom_point() +  
  geom_smooth()
```



Facets

- ▶ Instead of using colors to distinguish car class on one scatterplot, we can split into multiple scatterplots or “facets”.

```
ggplot(data=mpg, mapping = aes(x=displ,y=hwy,color=class)) +  
  geom_point() + geom_smooth() +  
  facet_wrap(~ class,nrow=2)
```



- ▶ (The colors are no longer necessary, but are included to emphasize how our earlier scatterplot has been split.)

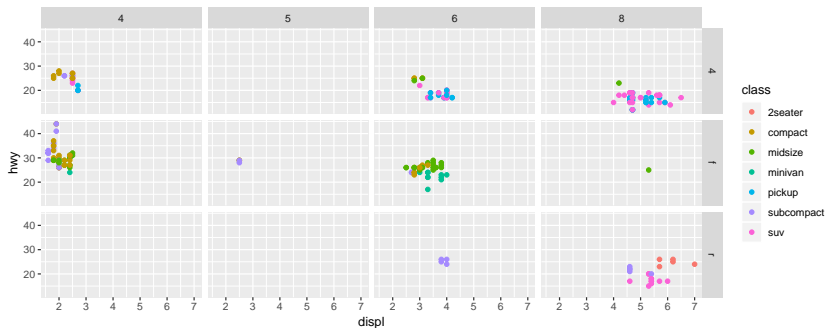
Faceting

- ▶ The facets by car class were added with `facet_wrap()`.
 - ▶ The variable to facet on was given in the “formula” ~ `class`. Formulas will make more sense in the next faceting example.
 - ▶ `facet_wrap()` would have put made three rows in this example. We force two with `nrow=2`.
- ▶ **Exercise:** Repeat the above example, but omit the `nrow=2` argument to `facet_wrap()`.

Faceting on two variables.

- ▶ The variable `drv` is 4-, front- or rear-wheel drive.
- ▶ The variable `cyl` is the number of cylinders.

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



ggplot layers

- ▶ The g's stand for Grammar of Graphics.
 - ▶ Like English grammar is the way in which words are put together to form sentences, a grammar of graphics is a way to put together basic graphical elements to make a graph.
- ▶ ggplots are built in layers, comprised of **data** a **mapping**, a **geom** and optionally **stats**, such as a scatterplot smoother.
- ▶ The layers are arranged and labelled on the graph by **scales** and **coords** (next lecture).
- ▶ The data can also be broken into subsets and displayed in separate graphs by a **facet** specification.

Another example: The gapminder data

- ▶ The gapminder dataset contains life expectancy, population size and GDP per capita for countries throughout the world from 1952 to 2007 in seven-year intervals.

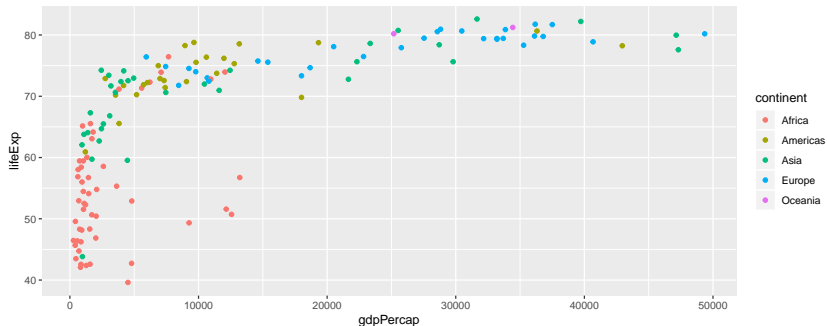
```
library(gapminder)
data(gapminder)
head(gapminder)
```

```
## # 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.
```

Subsetting the gapminder data

- Plot life expectancy *versus* GDP per capita for 2007.
 - Need to subset, or “filter” the data to observations from 2007.

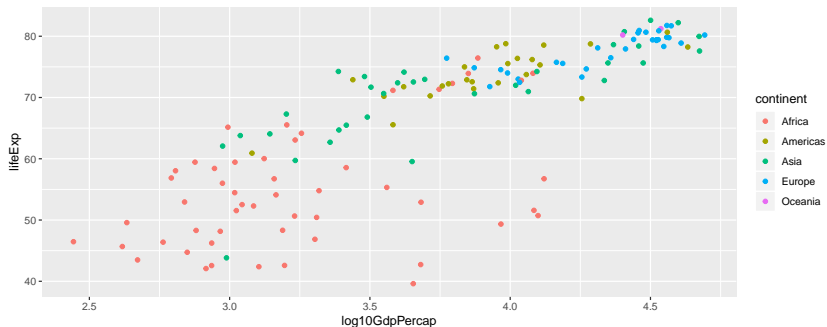
```
gm07 <- filter(gapminder, year==2007)
ggplot(gm07, aes(x=gdpPercap, y=lifeExp, color=continent)) +
  geom_point()
```



Transform GDP per capita

- ▶ GDP per capita differs by several orders of magnitude across countries. For such explanatory variables, the log scale may be more appropriate.

```
gm07 <- mutate(gm07, log10GdpPerCap = log10(gdpPerCap))  
ggplot(gm07, aes(x=log10GdpPerCap, y=lifeExp, color=continent)) +  
  geom_point()
```



Build a plot by layer

- ▶ In this example we build the plot by layer and do not show the plot until all layers are added.
- ▶ Our plot will be for the entire gapminder dataset.
- ▶ Set the mapping:

```
gapminder <- mutate(gapminder, log10GdpPerCap = log10(gdpPerCap))  
p <- ggplot(gapminder, aes(x=log10GdpPerCap, y=lifeExp, color=continent))
```

Add the geoms

- ▶ Overplotting means we can't tell how many points per area of the plot.
- ▶ Set a transparency, or “alpha” value to make points semi-transparent. Then many points will add up to solid, and few points will show as semi-transparent.

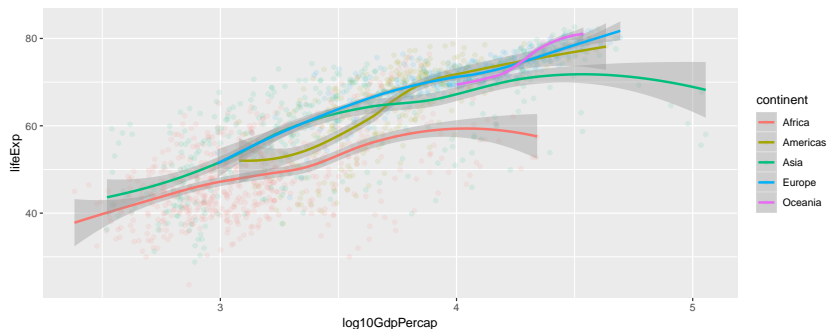
```
p2 <- p + geom_point(alpha=0.1)
```

- ▶ alpha is the transparency aesthetic, between 0 and 1, best applied directly to the geom.

Add statistical transformations

- Statistical transformations or **stats** summarize the data; e.g., a scatterplot smoother

```
p2 + stat_smooth()
```



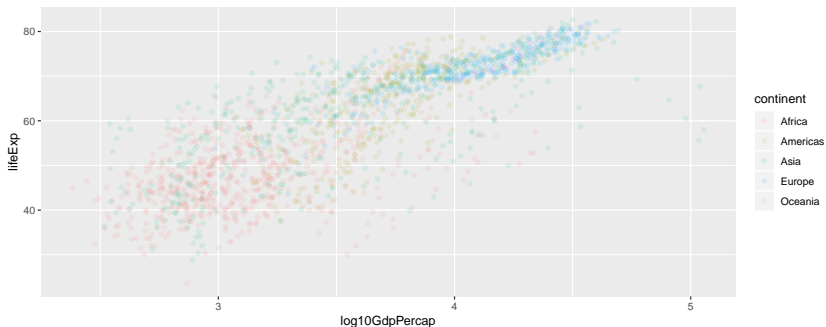
Exercise:

- ▶ The variable `year` is quantitative, but can still be used as a grouping variable. Do scatterplots of `lifeExp` versus `log10GdpPercap` with points colored by `year`. Add a scatterplot smoother with (i) no grouping variable and (ii) `year` as the grouping variable. Set the `SE` to `FALSE` for your smoothers.

ggplot scales

- ▶ The **scales** are mappings from the data to the graphics device
 - ▶ domain of continent is the five continents, range is the hexadecimal of the five colors represented on the graph
 - ▶ domain of lifeExp is 23.599 to 82.603, range is [0,1], which grid converts to a range of vertical pixels on the graph.
 - ▶ legends and axes provide the inverse mapping

p2



ggplot coordinate system

- ▶ The coordinate system is another layer in how the data get mapped to the graphics device.
 - ▶ Usually Cartesian, but could be, e.g., polar coordinates, or a transformation.

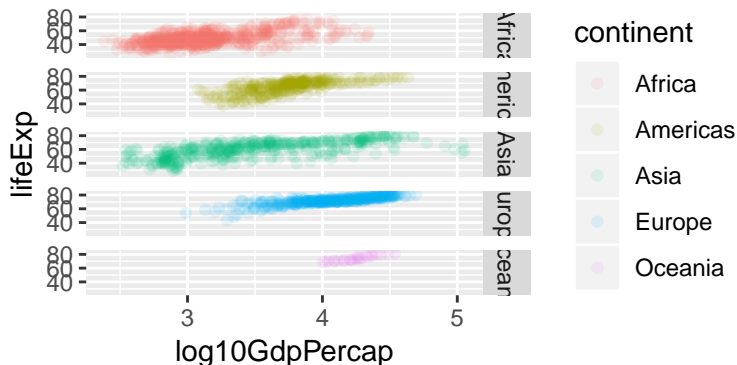
```
ggplot(gapminder, aes(x=gdpPercap, y=lifeExp, color=continent)) +  
  geom_point(alpha=0.5) + coord_trans(x="log10")
```



ggplot faceting

- How to break up the data into subsets and arrange multiple plots on the graphics device.

```
p2 + facet_grid(continent ~ .)
```



Why so many components?

- ▶ A framework for the components of a graph.
- ▶ Gives the user the ability to change individual components one at a time.