Introduction to Applied Data Science

Bas Machielsen
Utrecht University

2022-10-21

Elements of ggplot2

- Hadley Wickham's ggplot2 is one of the most popular packages in the entire R canon.
- It also happens to be built upon some deep visualization theory: i.e. Leland Wilkinson's *The Grammar of Graphics*.

There's a lot to say about ggplot2's implementation of this "grammar of graphics" approach, but the three key elements are:

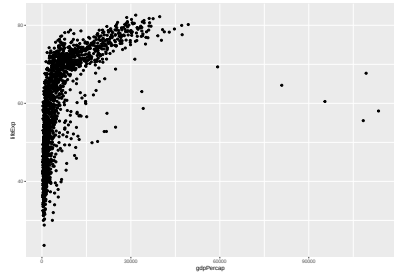
1. Your plot ("the visualization") is linked to your variables ("the data") through various **aesthetic mappings**.
2. Once the aesthetic mappings are defined, you can represent your data in different ways by choosing different **geoms** (i.e. "geometric objects" like points, lines or bars).
3. You build your plot in **layers**.

- That's kind of abstract. Let's review each element in turn with some actual plots.

1. Aesthetic mappings

```
pacman::p_load(tidyverse, gapminder)

ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp)) +
  geom_point()
```



1. Aesthetic mappings (cont.)

```
ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp)) +
  geom_point()
```

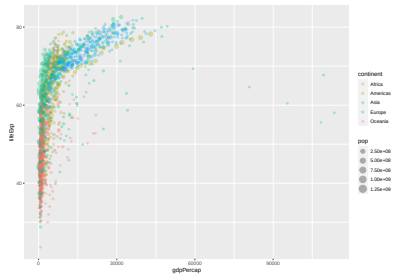
- Focus on the top line, which contains the initialising `ggplot()` function call. This function accepts various arguments, including:
  - Where the data come from (i.e. `data = gapminder`).
  - What the aesthetic mappings are (i.e. `mapping = aes(x = gdpPercap, y = lifeExp)`).

1. Aesthetic mappings (cont.)

- The aesthetic mappings here are pretty simple: They just define an x-axis (GDP per capita) and a y-axis (life expecancy).
- To get a sense of the power and flexibility that comes with this approach, however, consider what happens if we add more aesthetics to the plot call...

1. Aesthetic mappings (cont.)

```
ggplot(data = gapminder, aes(x = gdpPercap, y = lifeExp, size = pop, col = continent)) +
  geom_point(alpha = 0.3) ## "alpha" controls transparency. Takes a value between 0 and 1.
```

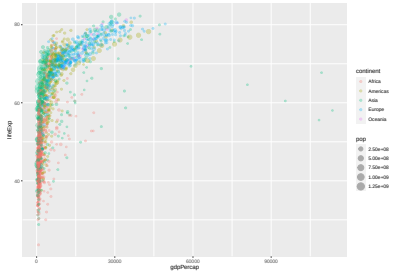


- Note that I've dropped the "mapping =" part of the ggplot call. Most people just start with "aes(...)", since ggplot2 knows the order of the arguments.

1. Aesthetic mappings (cont.)

- We can specify aesthetic mappings in the geom layer too.

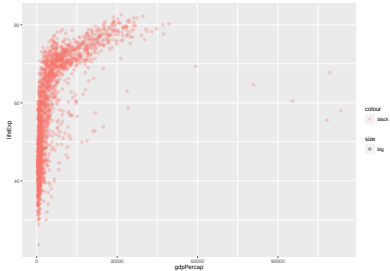
```
ggplot(data = gapminder, aes(x = gdpPercap, y = lifeExp)) + ## Applicable to all geoms
  geom_point(aes(size = pop, col = continent), alpha = 0.3) ## Applicable to this geom only
```



1. Aesthetic mappings (cont.)

- Oops. What went wrong here?

```
ggplot(data = gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point(aes(size = "big", col="black"), alpha = 0.3)
```

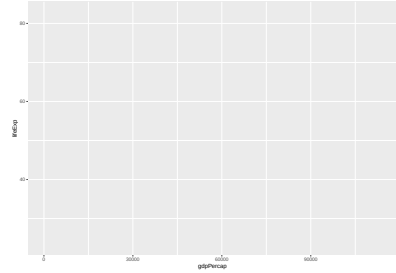


Answer: Aesthetics must be mapped to variables, not descriptions!

1. Aesthetic mappings (cont.)

- At this point, instead of repeating the same ggplot2 call every time, it will prove convenient to define an intermediate plot object that we can re-use.

```
p = ggplot(data = gapminder, aes(x = gdpPercap, y = lifeExp))
p
```

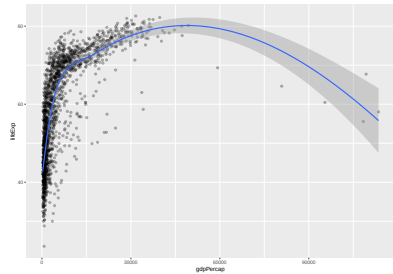


2. Geoms

- Once your variable relationships have been defined by the aesthetic mappings, you can invoke and combine different geoms to generate different visualizations.

```
p +
  geom_point(alpha = 0.3)  +
  geom_smooth(method = "loess")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

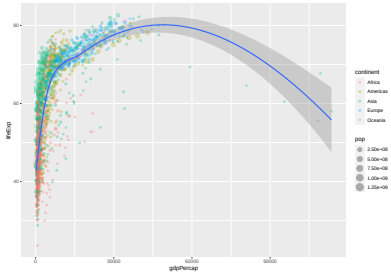


2. Geoms (cont.)

- Aesthetics can be applied differentially across geoms.

```
p +
  geom_point(aes(size = pop, col = continent), alpha = 0.3)  +
  geom_smooth(method = "loess")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



2. Geoms (cont.)

- The previous plot provides a good illustration of the power (or effect) that comes from assigning aesthetic mappings "globally" vs in the individual geom layers.
- Compare: What happens if you run the below code chunk?

```
ggplot(data = gapminder, aes(x = gdpPercap, y = lifeExp, size = pop, col = continent)) +
  geom_point(alpha = 0.3)  +
  geom_smooth(method = "loess")
```

2. Geoms (cont.)

- Similarly, note that some geoms only accept a subset of mappings. E.g. `geom_density()` doesn't know what to do with the "y" aesthetic mapping.

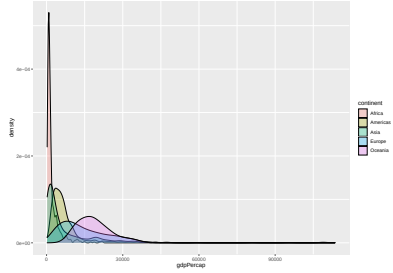
```
p + geom_density()
```

```
## Error in `check_required_aesthetics()`:
## ! geom_density requires the following missing aesthetics: y
```

2. Geoms (cont.)

- We can fix that by being more careful about how we build the plot.

```
ggplot(data = gapminder) + ## i.e. No "global" aesthetic mappings"
  geom_density(aes(x = gdpPercap, fill = continent), alpha=0.3)
```



3. Build your plot in layers

- We've already seen how we can chain (or "layer") consecutive plot elements using the + connector.
  - The fact that we can create and then re-use an intermediate plot object (e.g. "p") is testament to this.
- But it bears repeating: You can build out some truly impressive complexity and transformation of your visualization through this simple layering process.
  - You don't have to transform your original data; ggplot2 takes care of all of that.
  - For example (see next slide for figure).

```
p2 =
  p +
  geom_point(aes(size = pop, col = continent), alpha = 0.3) +
  scale_color_brewer(name = "Continent", palette = "Set1") + ## Different color scale
  scale_size(name = "Population", labels = scales::comma) + ## Different point (i.e. legend
  scale_x_log10(labels = scales::dollar) + ## Switch to logarithmic scale on x-axis. Use do
  labs(x = "Log (GDP per capita)", y = "Life Expectancy") + ## Better axis titles
  theme_minimal() ## Try a minimal (b&w) plot theme
```

3. Build your plot in layers (cont.)