

Make Some Graphs

Modern Plain Text Social Science

Week 03

Kieran Healy

Duke University

September 2026

Make Some Graphs

Load our libraries

```
library(here)      # manage file paths
library(socviz)    # data and some useful functions
library(tidyverse) # your friend and mine
```

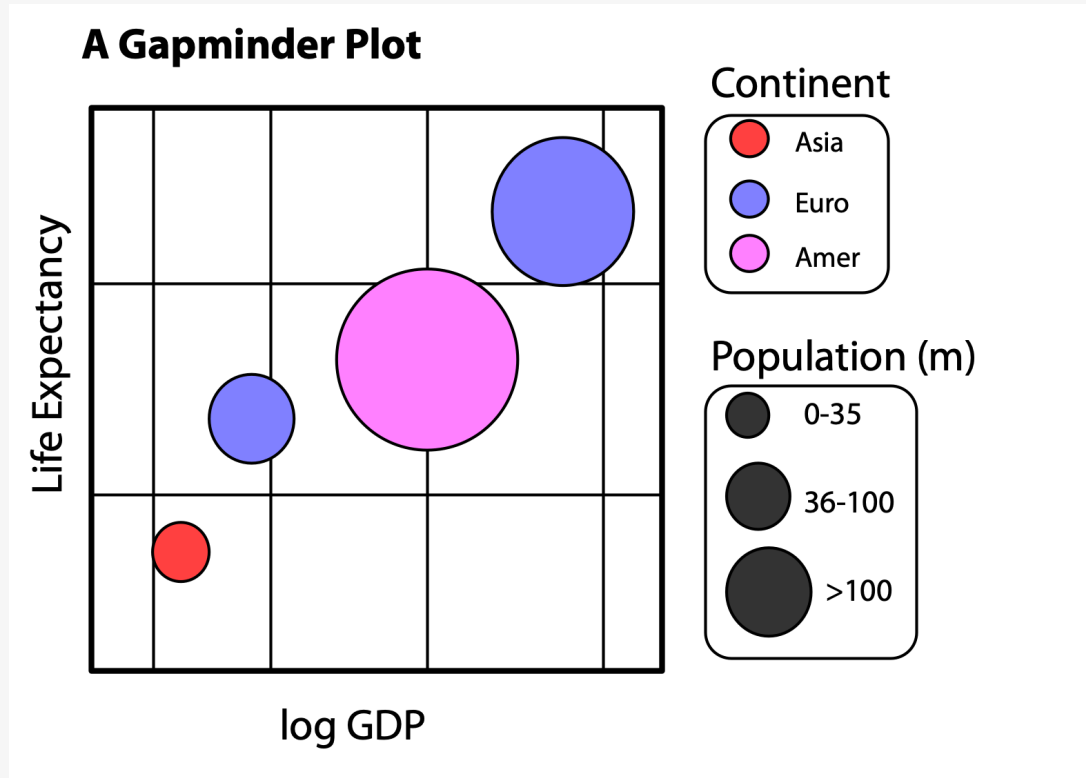
```
— Attaching core tidyverse packages — tidyverse 2.0.0 —
✓ dplyr      1.2.1      ✓ readr      2.2.0
✓ forcats    1.0.1      ✓ stringr    1.6.0
✓ ggplot2     4.0.3      ✓ tibble     3.3.1
✓ lubridate  1.9.5      ✓ tidyr      1.3.2
✓ purrr       1.2.2

— Conflicts — tidyverse_conflicts() —
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()     masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(gapminder) # some data
```

A Plot's Components

What we need our code to make



Data **represented** by visual elements;

like *position*, *length*, *color*, and *size*;

Each measured on some **scale**;

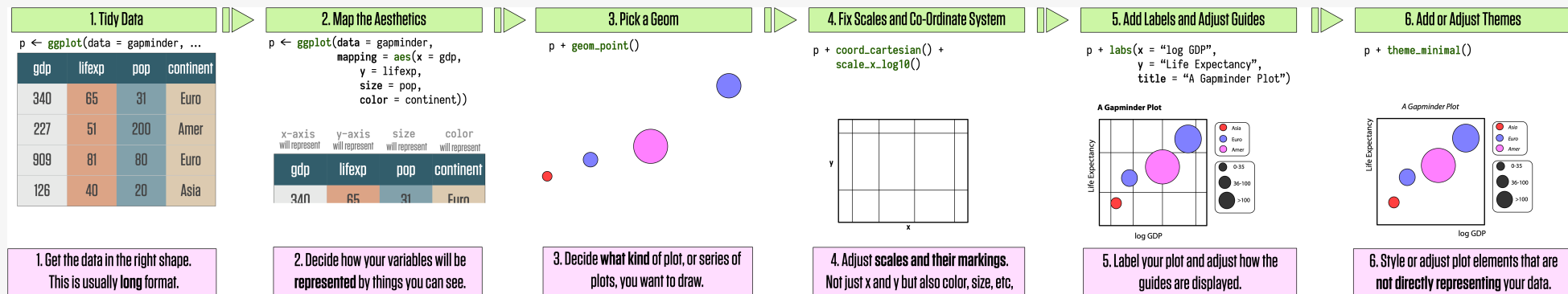
Each scale with a labeled **guide**;

With the plot itself also **titled** and labeled.

How does
ggplot
do this?

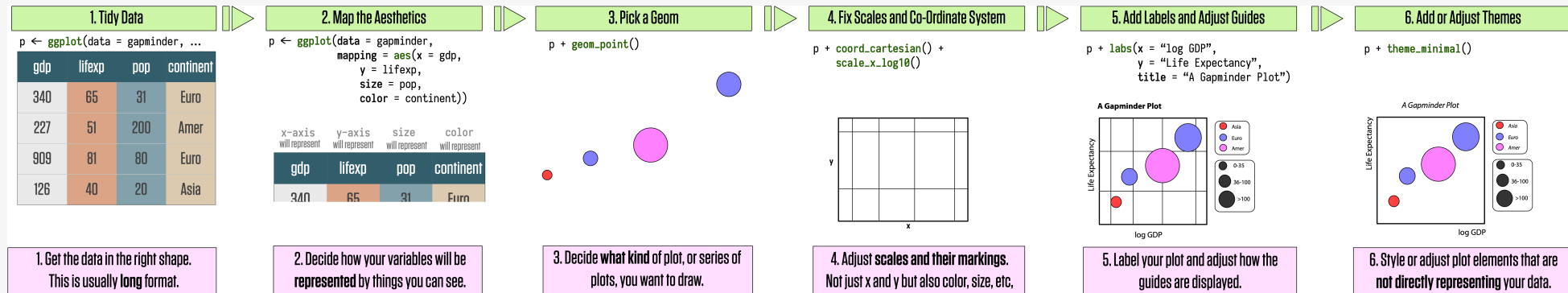
ggplot's flow of action

Here's the whole thing, start to finish



Flow of action

We'll go through it step by step



Flow of action

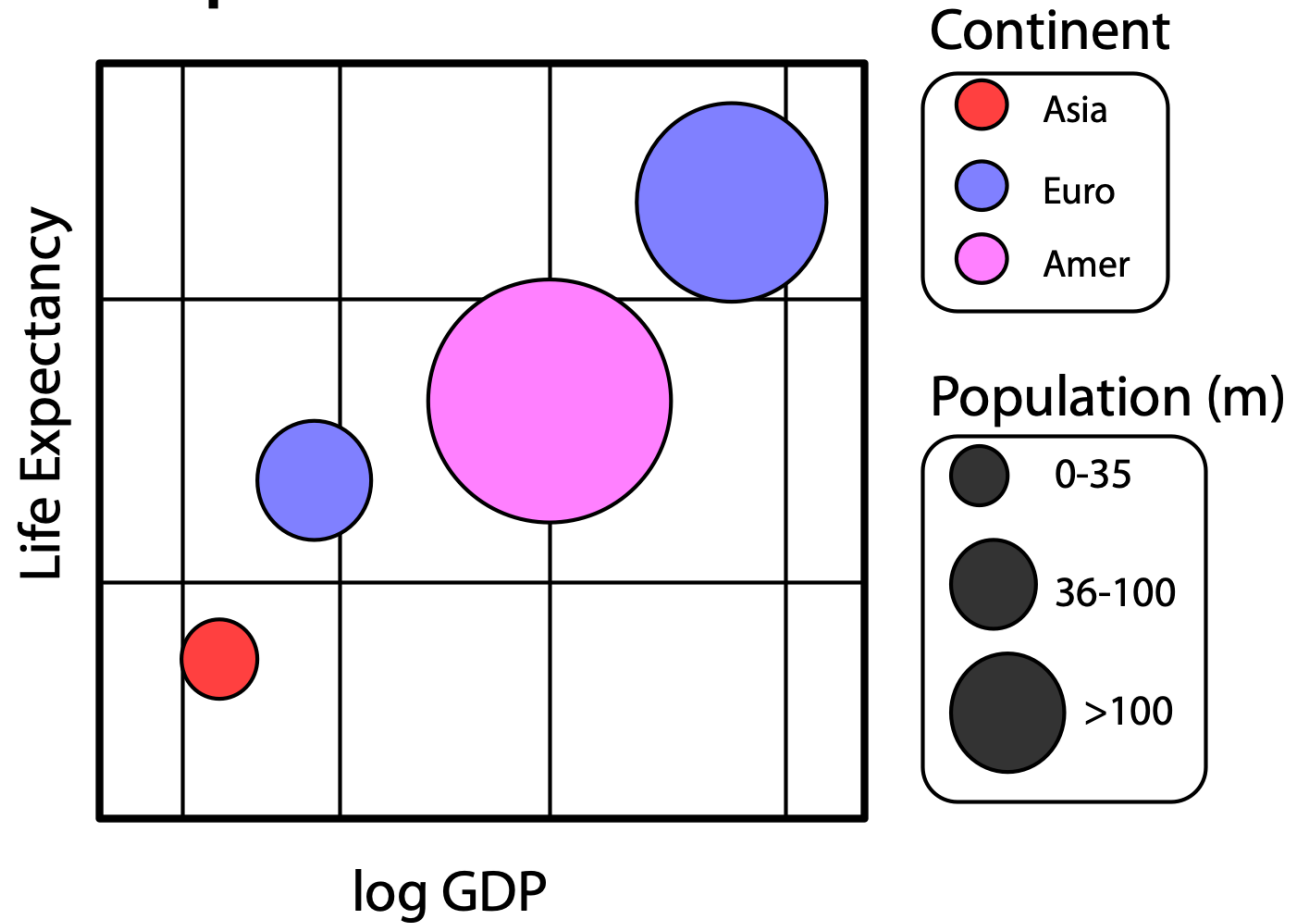
ggplot's flow of action

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

What we start with

ggplot's flow of action

A Gapminder Plot



Where we're going

ggplot's flow of action

1. Tidy Data

```
p <- ggplot(data = gapminder, ...
```

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

1. Get the data in the right shape.
This is usually **long** format.

2. Map the Aesthetics

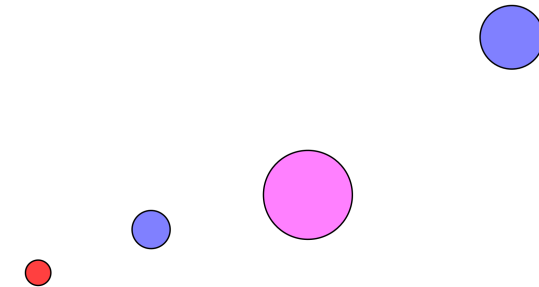
```
p <- ggplot(data = gapminder,  
  mapping = aes(x = gdp,  
    y = lifexp,  
    size = pop,  
    color = continent))
```

x-axis will represent	y-axis will represent	size will represent	color will represent
gdp	lifexp	pop	continent
340	65	31	Euro

2. Decide how your variables will be
represented by things you can see.

3. Pick a Geom

```
p + geom_point()
```



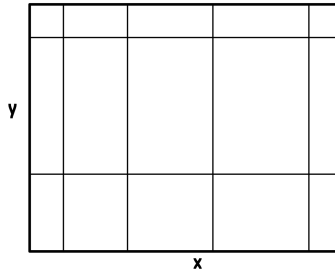
3. Decide **what kind** of plot, or series of
plots, you want to draw.

Core steps

ggplot's flow of action

4. Fix Scales and Co-Ordinate System

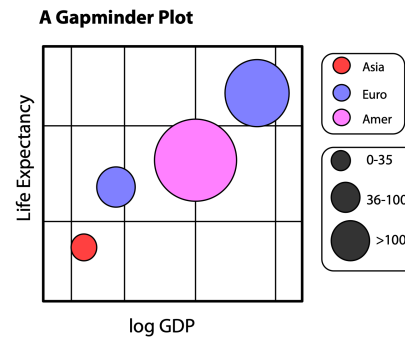
```
p + coord_cartesian() +  
  scale_x_log10()
```



4. Adjust **scales and their markings**.
Not just x and y but also color, size, etc,

5. Add Labels and Adjust Guides

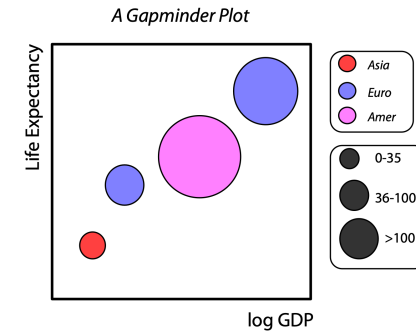
```
p + labs(x = "log GDP",  
        y = "Life Expectancy",  
        title = "A Gapminder Plot")
```



5. Label your plot and adjust how the
guides are displayed.

6. Add or Adjust Themes

```
p + theme_minimal()
```



6. Style or adjust plot elements that are
not directly representing your data.

Optional steps

ggplot's flow of action: required

1. Tidy Data

```
p ← ggplot(data = gapminder, ...)
```

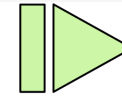
gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

1. Get the data in the right shape.
This is usually **long** format.

Tidy data

ggplot's flow of action: required

2. Map the Aesthetics



```
p <- ggplot(data = gapminder,  
            mapping = aes(x = gdp,  
                          y = lifexp,  
                          size = pop,  
                          color = continent))
```

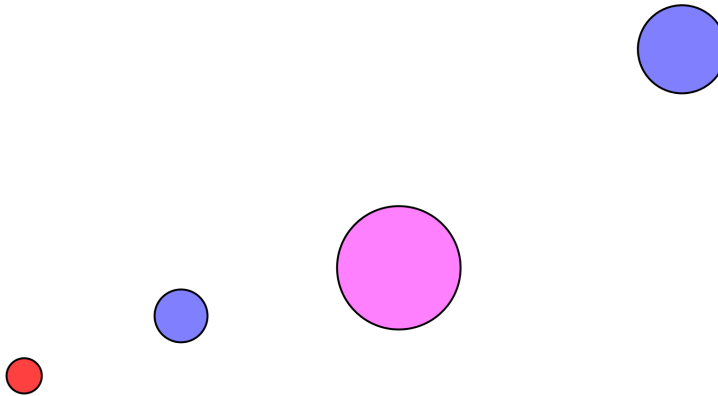
x-axis will represent	y-axis will represent	size will represent	color will represent
gdp	lifexp	pop	continent
340	65	31	Euro

2. Decide how your variables will be represented by things you can see.

ggplot's flow of action: required

3. Pick a Geom

```
p + geom_point()
```



3. Decide what kind of plot, or series of plots, you want to draw.

**Let's go piece by
piece**

Start with the data

```
gapminder
```

```
# A tibble: 1,704 × 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.
7	Afghanistan	Asia	1982	39.9	12881816	978.
8	Afghanistan	Asia	1987	40.8	13867957	852.
9	Afghanistan	Asia	1992	41.7	16317921	649.
10	Afghanistan	Asia	1997	41.8	22227415	635.

```
# i 1,694 more rows
```

```
dim(gapminder)
```

```
[1] 1704    6
```

Create a plot object

Data is the `gapminder` tibble.

```
p ← ggplot(data = gapminder)
```

Map variables to aesthetics

Tell **ggplot** the variables you want represented by visual elements on the plot

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

Map variables to aesthetics

The `mapping = aes( ... )` call links variables to things you will see on the plot.

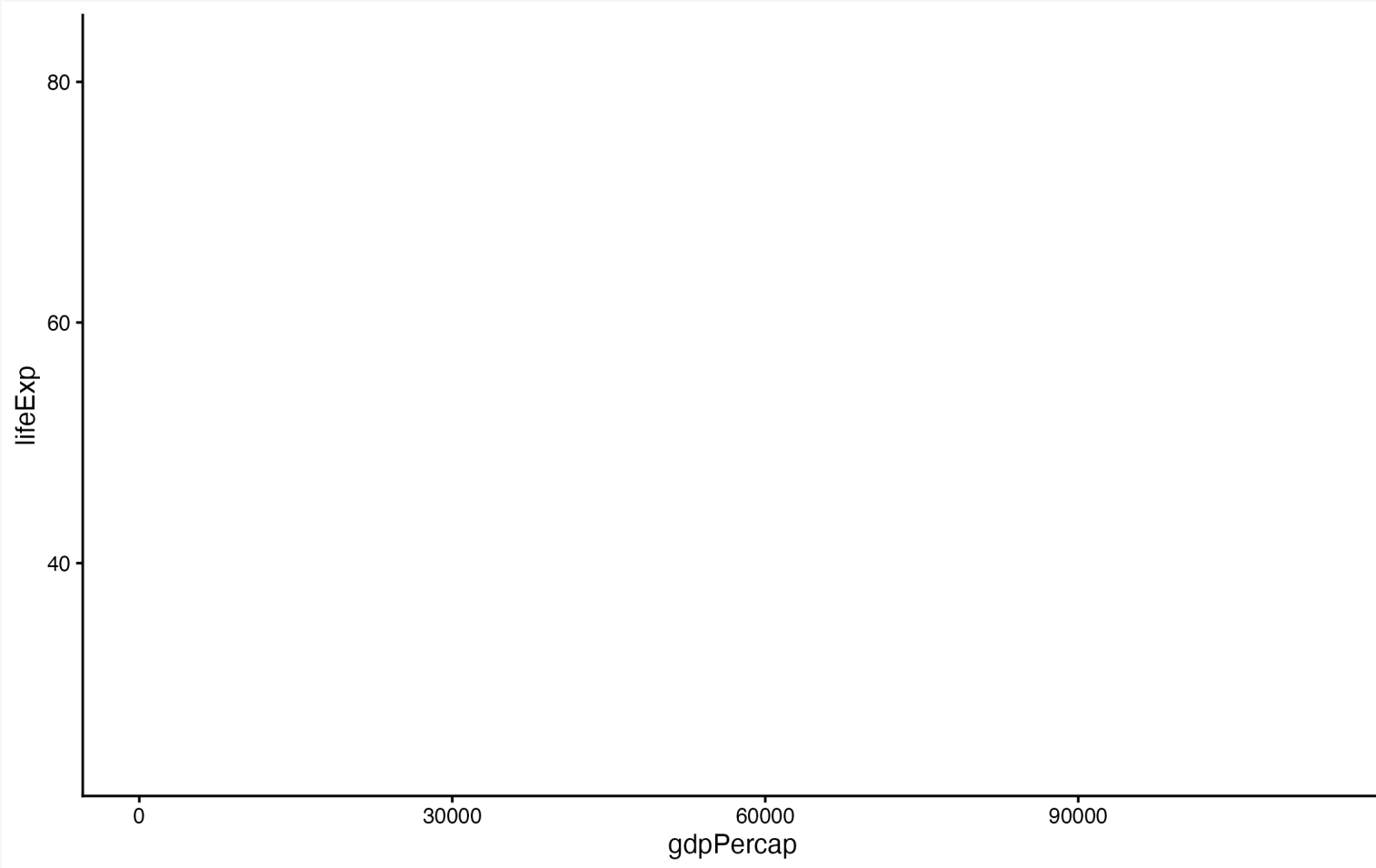
`x` and `y` represent the quantities determining position on the x and y axes.

Other aesthetic mappings can include, e.g., `color`, `shape`, `size`, and `fill`.

Mappings do not *directly* specify the particular, e.g., colors, shapes, or line styles that will appear on the plot. Rather, they establish *which variables* in the data will be represented by *which visible elements* on the plot.

p has data and mappings but no geom

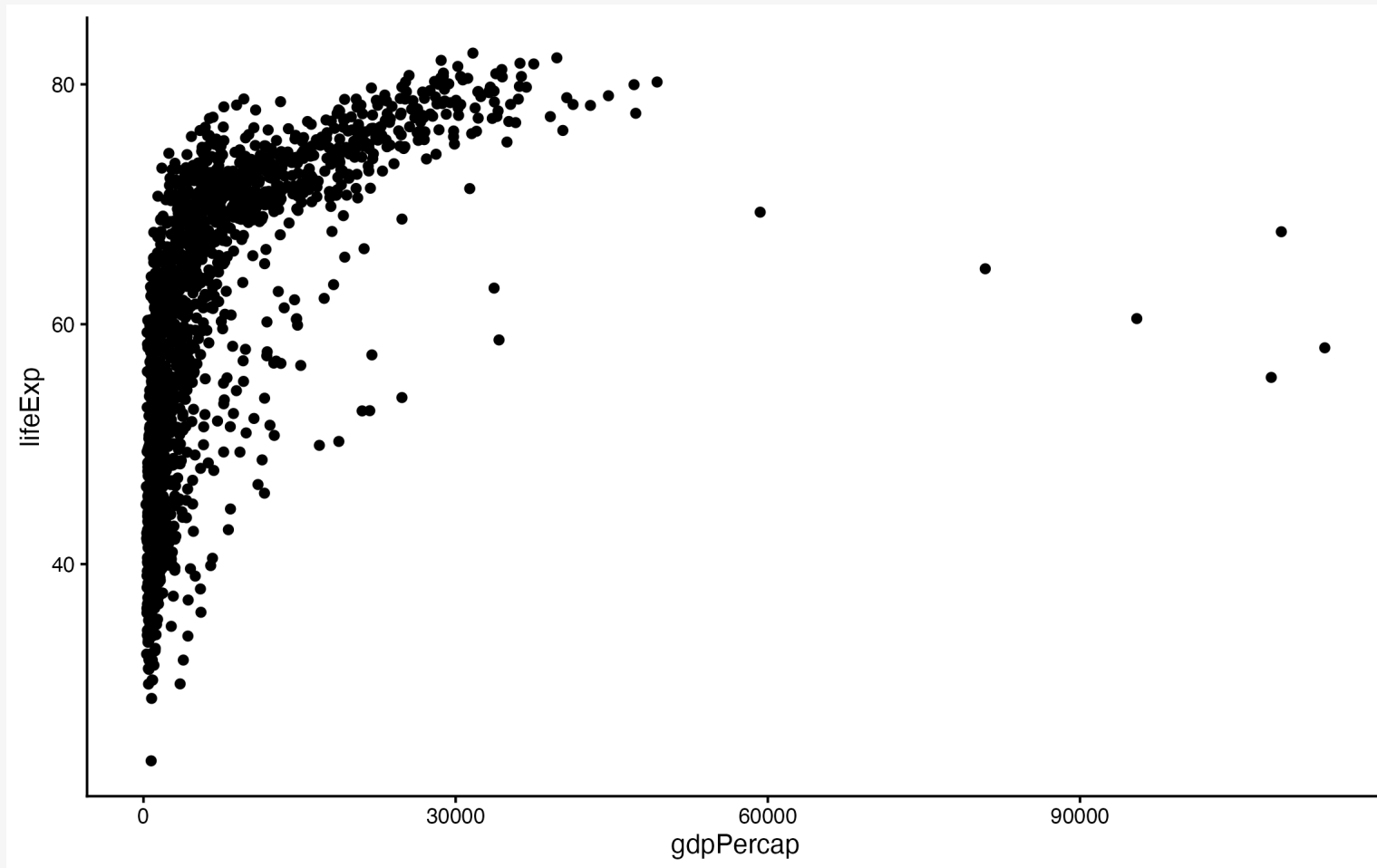
p



This empty plot has no geoms.

Add a geom

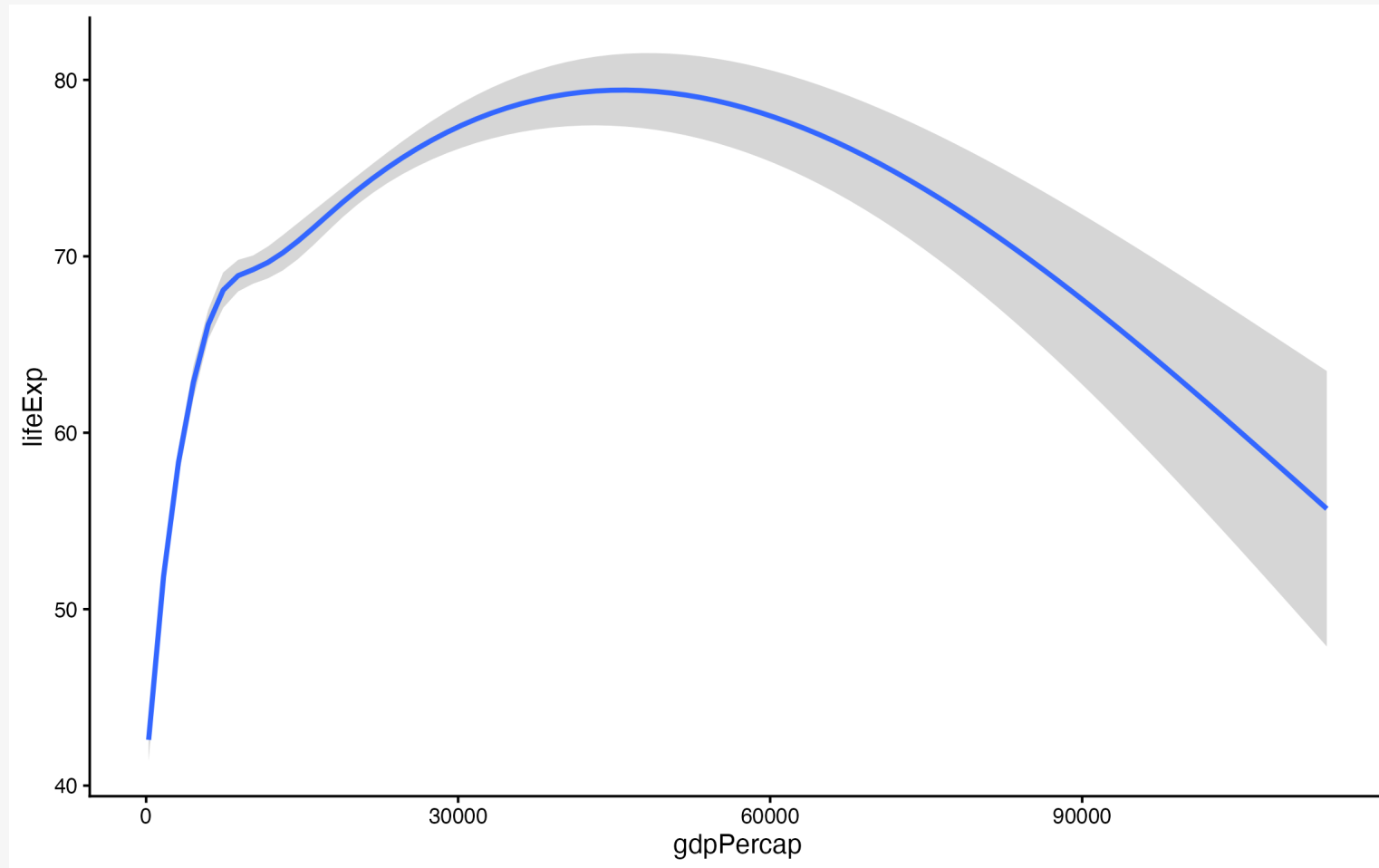
```
p + geom_point()
```



A scatterplot of Life Expectancy vs GDP

Try a different geom

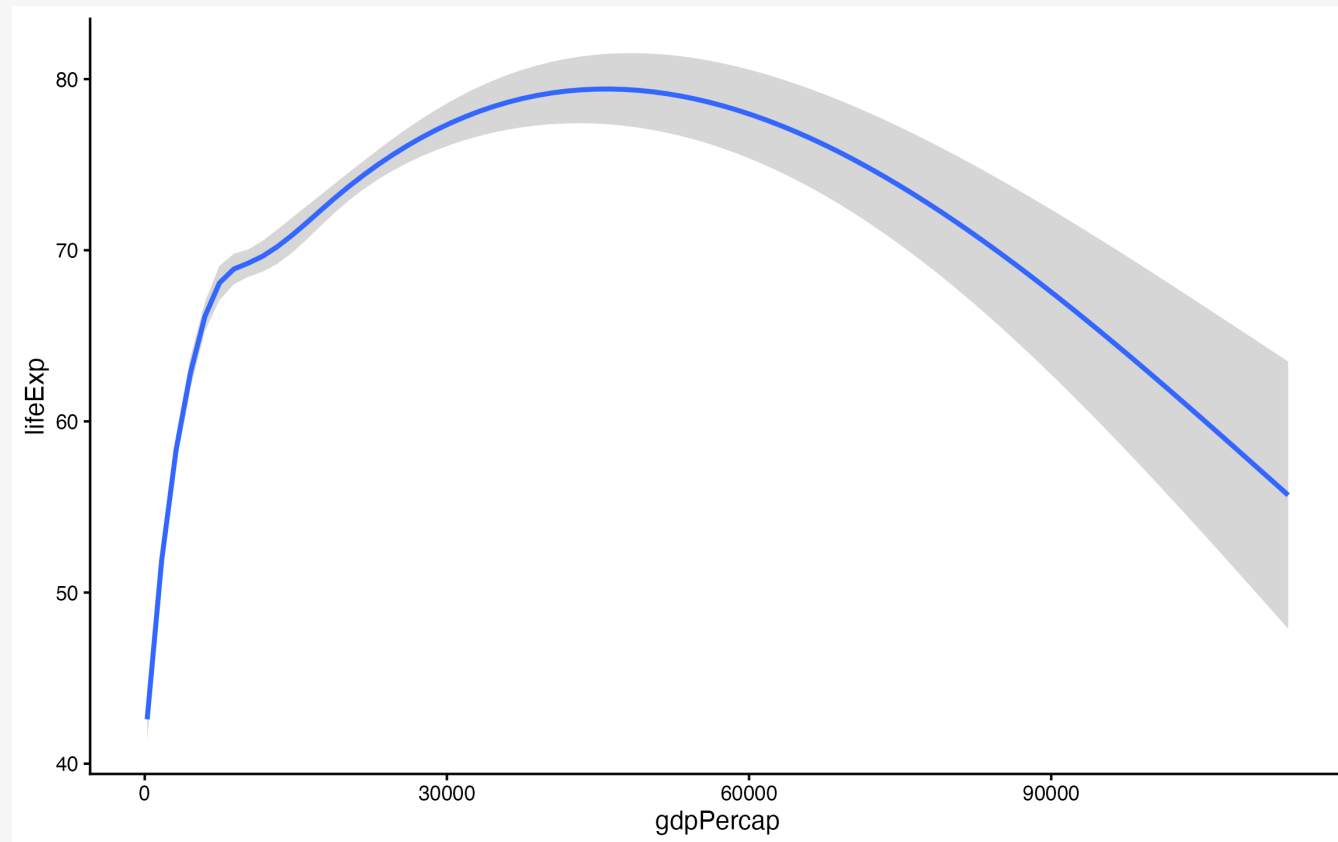
```
p + geom_smooth()
```



A scatterplot of Life Expectancy vs GDP

Build your plots layer by layer

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y=lifeExp))  
p + geom_smooth()
```



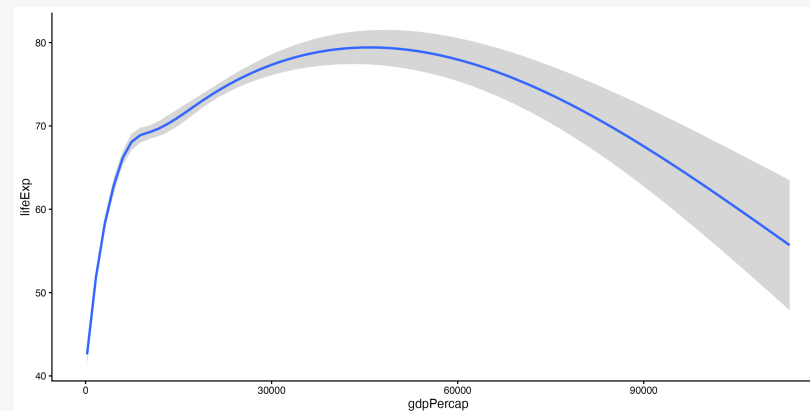
Life Expectancy vs GDP, using a smoother.

This process is additive

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

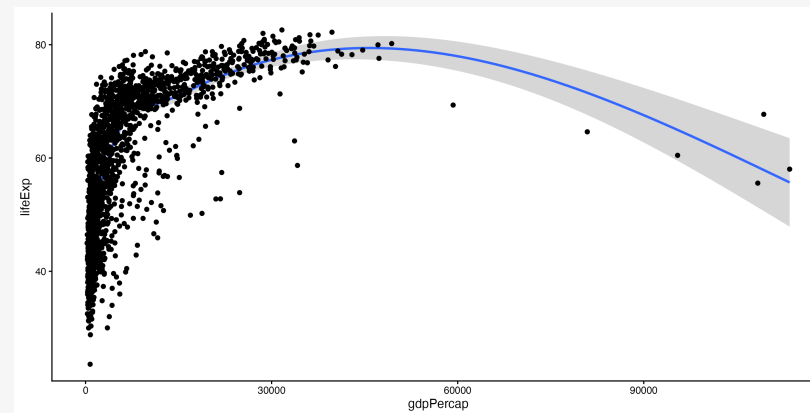
This process is additive

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y=lifeExp))  
p + geom_smooth()
```



This process is additive

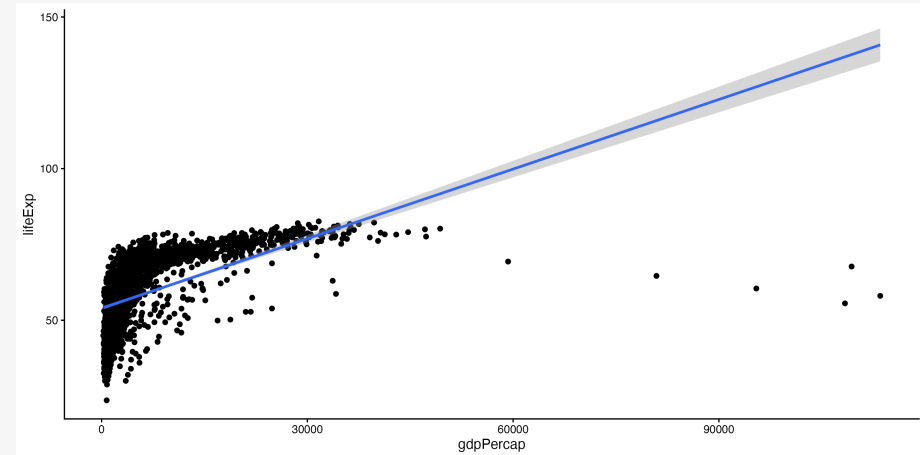
```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y=lifeExp))  
  
p + geom_smooth() +  
  geom_point()
```



Every geom is a function

Functions take arguments

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y = lifeExp))  
p + geom_point() +  
  geom_smooth(method = "lm")
```

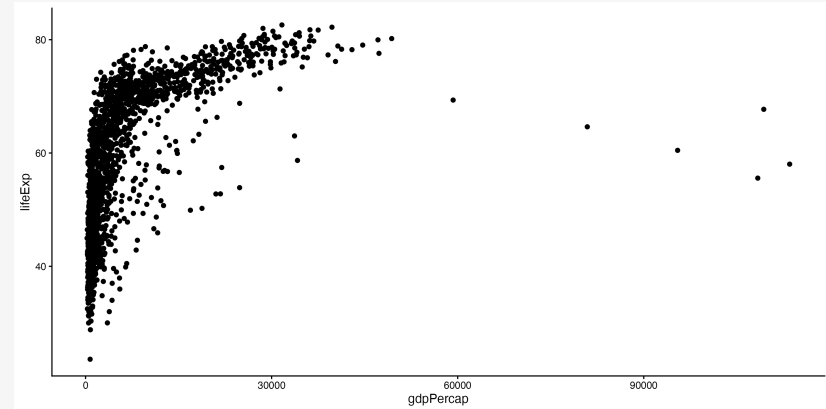


Keep Layering

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

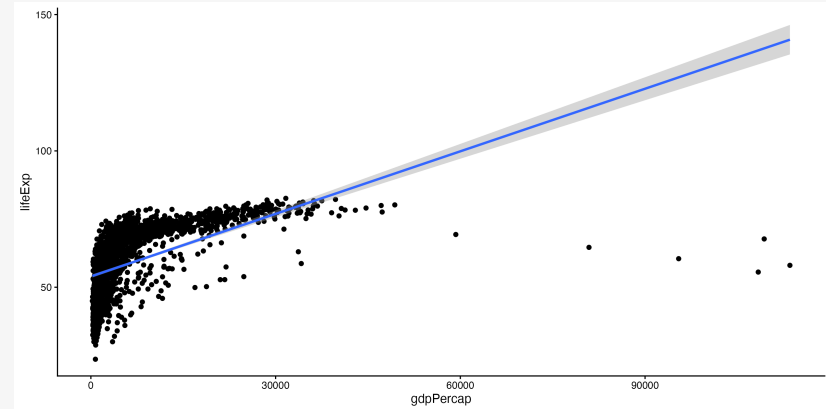
Keep Layering

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



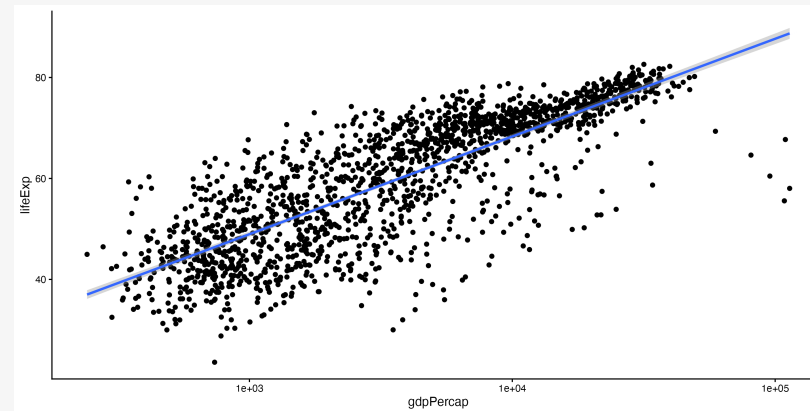
Keep Layering

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y=lifeExp))  
p + geom_point() +  
  geom_smooth(method = "lm")
```



Keep Layering

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y=lifeExp))  
  
p + geom_point() +  
  geom_smooth(method = "lm") +  
  scale_x_log10()
```

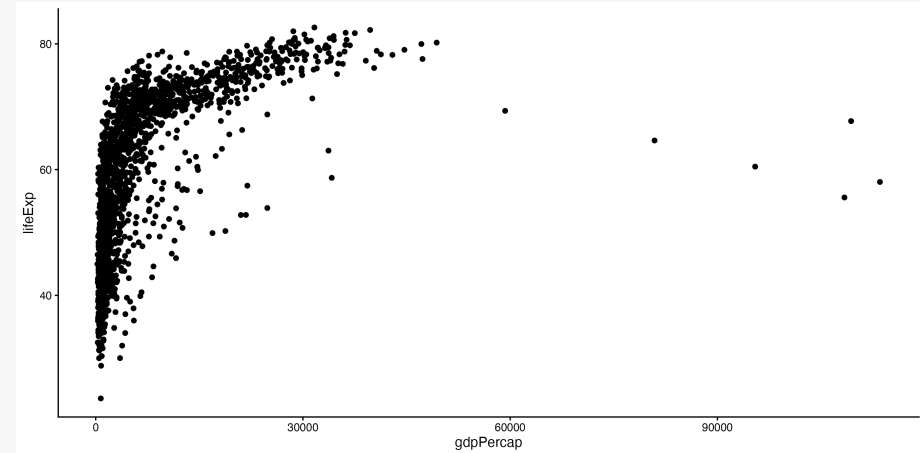


Fix the labels

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

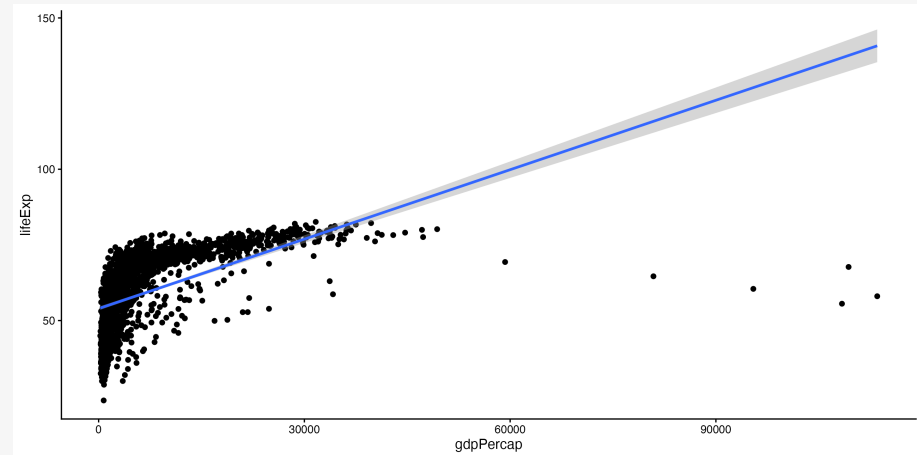
Fix the labels

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



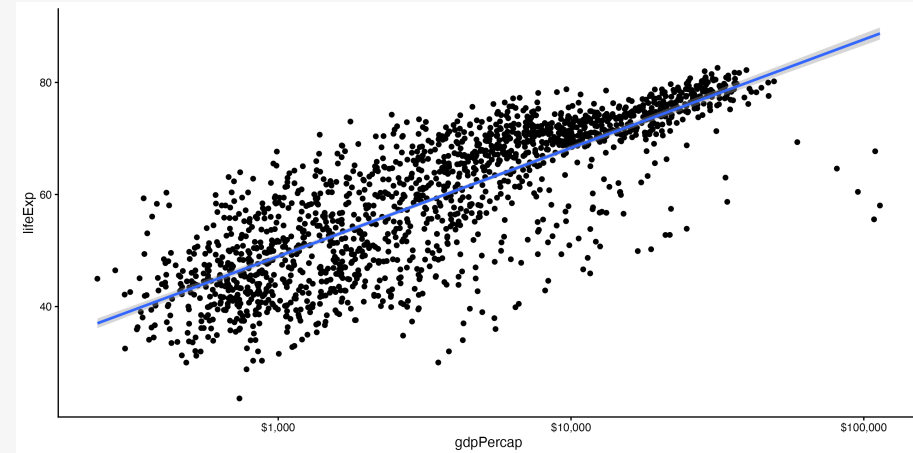
Fix the labels

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y=lifeExp))  
p + geom_point() +  
  geom_smooth(method = "lm")
```



Fix the labels

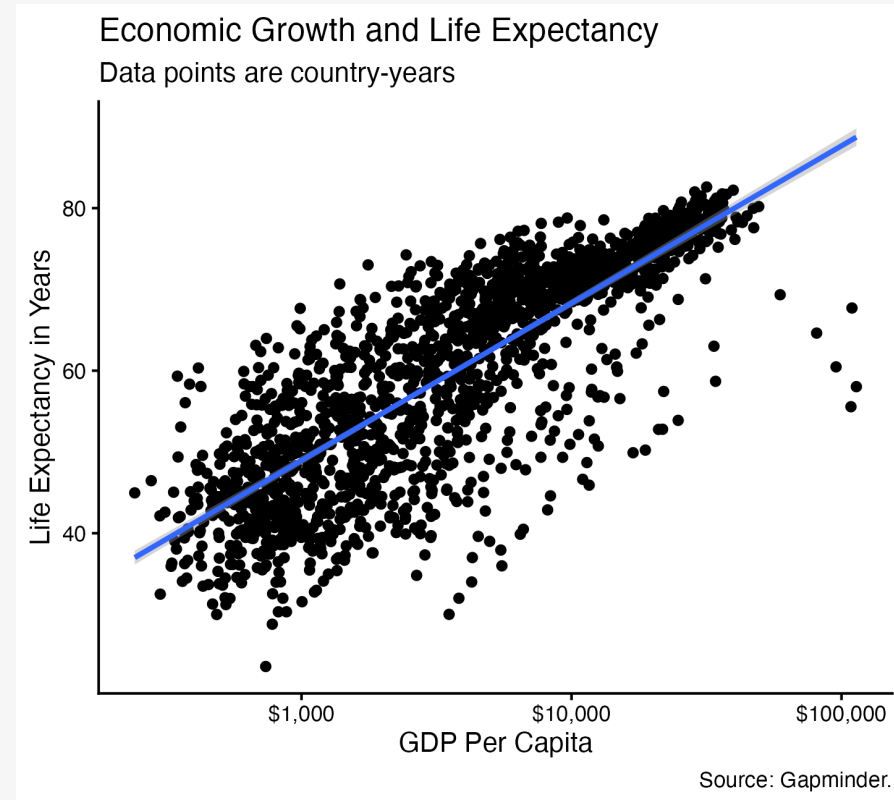
```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y=lifeExp))  
  
p + geom_point() +  
  geom_smooth(method = "lm") +  
  scale_x_log10(labels = scales::label_dollar())
```



Add labels, title, and caption

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

p + geom_point() +
  geom_smooth(method = "lm") +
  scale_x_log10(labels = scales::label_dollar()) +
  labs(x = "GDP Per Capita",
       y = "Life Expectancy in Years",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



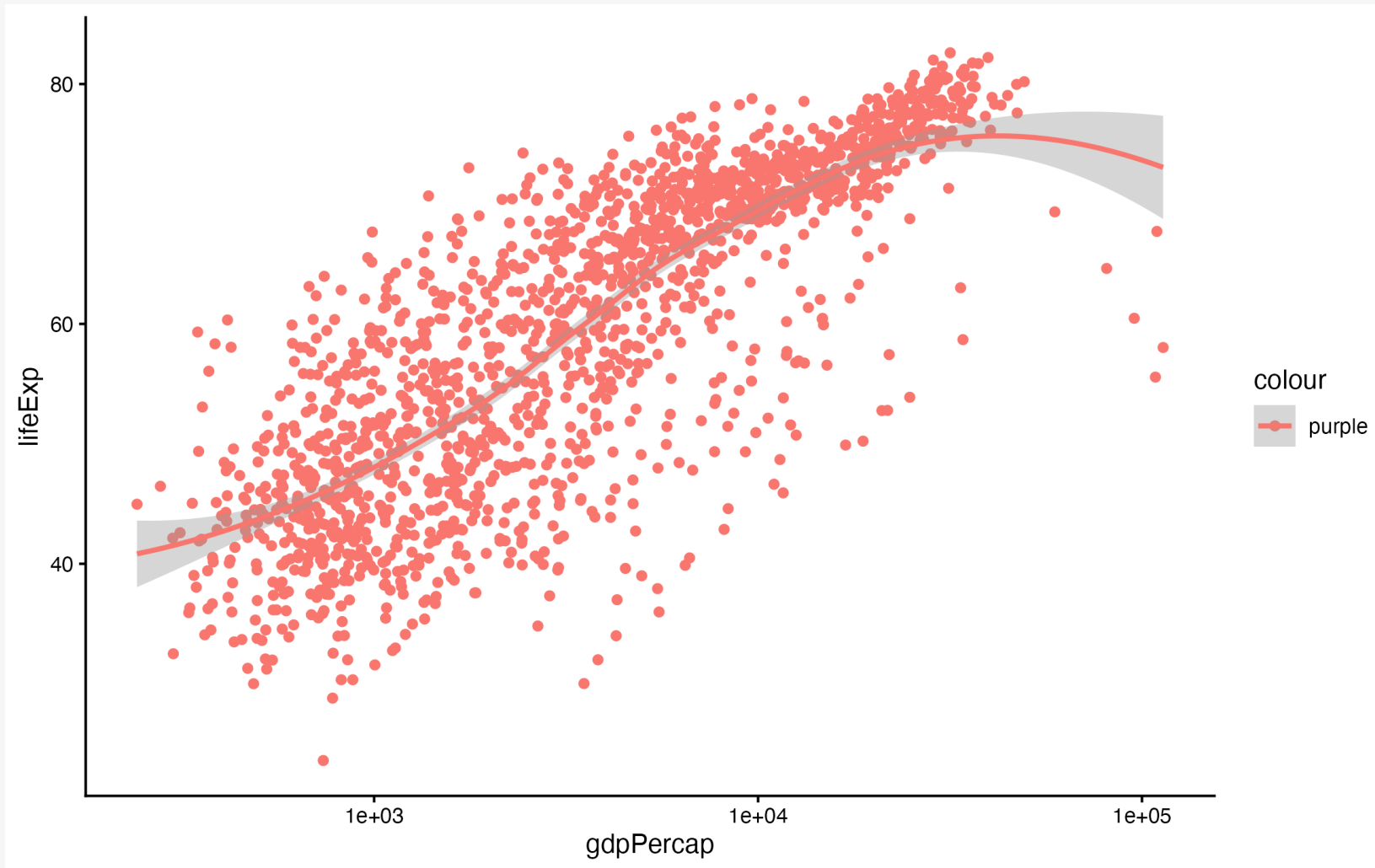
Mapping vs Setting your plot's aesthetics

“Can I change the color of the points?”

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y = lifeExp,  
                         color = "purple"))  
  
## Put in an object for convenience  
p_out ← p + geom_point() +  
          geom_smooth(method = "loess") +  
          scale_x_log10()
```

What has gone wrong here?

p_out

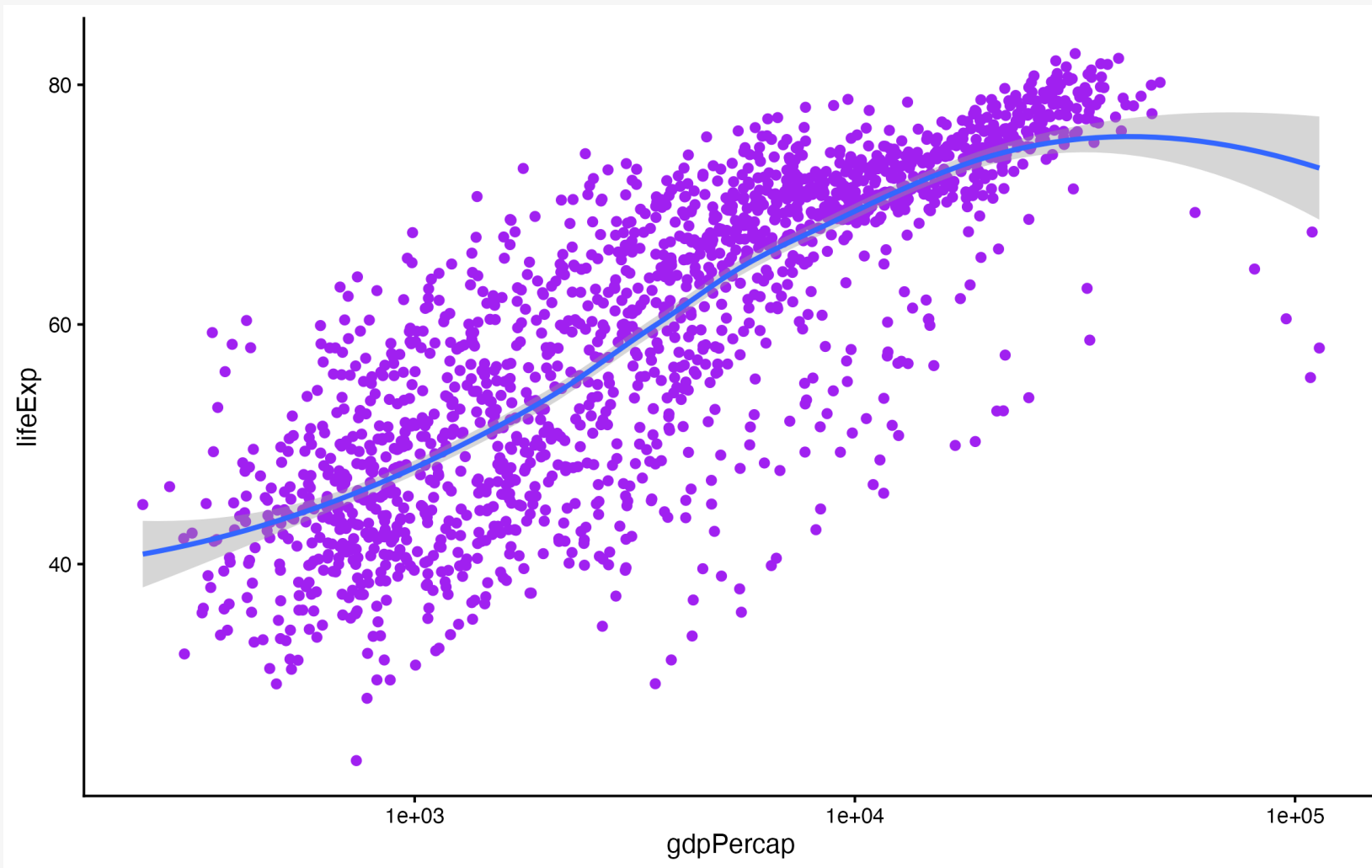


Try again

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y = lifeExp))  
  
## Put in an object for convenience  
p_out ← p + geom_point(color = "purple") +  
          geom_smooth(method = "loess") +  
          scale_x_log10()
```

Try again

p_out



Geoms can take many arguments

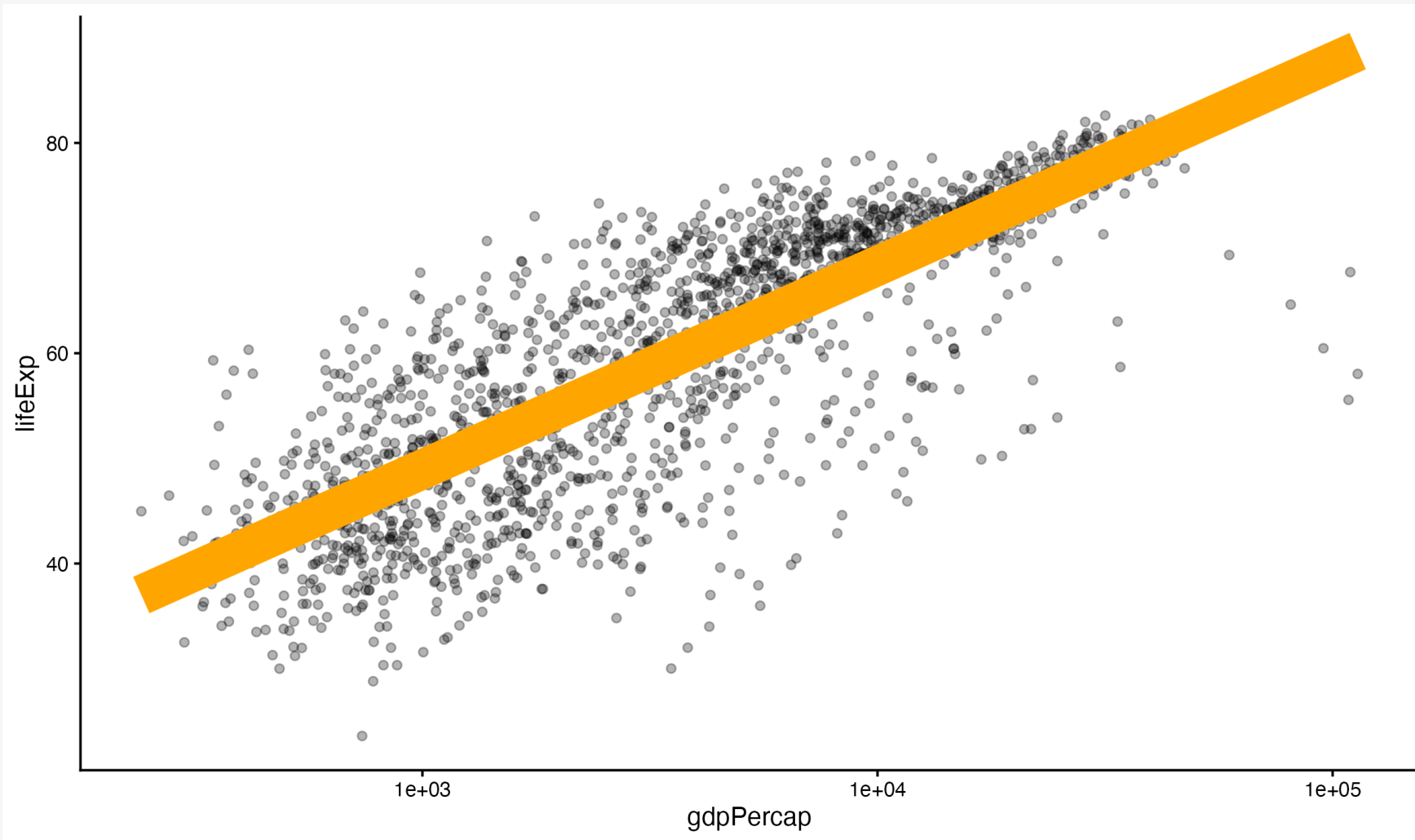
Here we **set** **color**, **size**, and **alpha**. Meanwhile **x** and **y** are **mapped**.

We also give non-default values to some other arguments

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y = lifeExp))  
p_out ← p + geom_point(alpha = 0.3) +  
  geom_smooth(color = "orange",  
             se = FALSE,  
             linewidth = 8,  
             method = "lm") +  
  scale_x_log10()
```

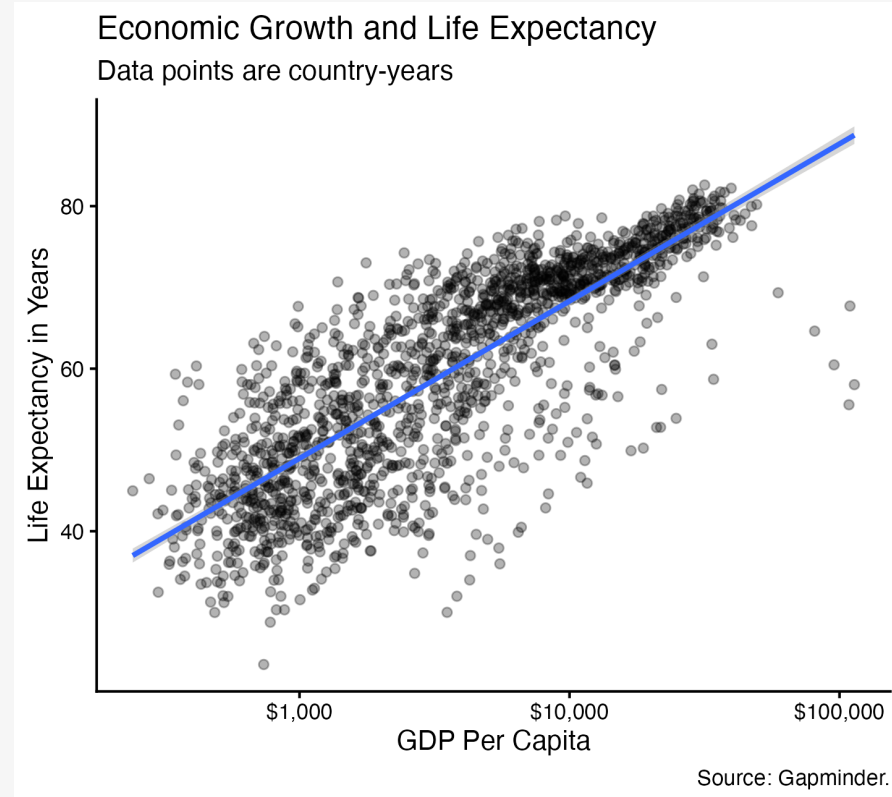
Geoms can take many arguments

p_out



alpha for overplotting

```
p ← ggplot(data = gapminder,
           mapping = aes(x = gdpPercap,
                        y = lifeExp))
p + geom_point(alpha = 0.3) +
  geom_smooth(method = "lm") +
  scale_x_log10(labels = scales::label_dollar()) +
  labs(x = "GDP Per Capita",
       y = "Life Expectancy in Years",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```



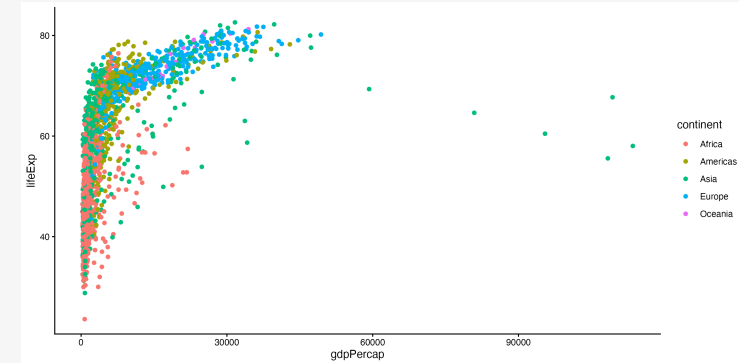
Map or Set values
per geom

Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y = lifeExp,  
                         color = continent,  
                         fill = continent))
```

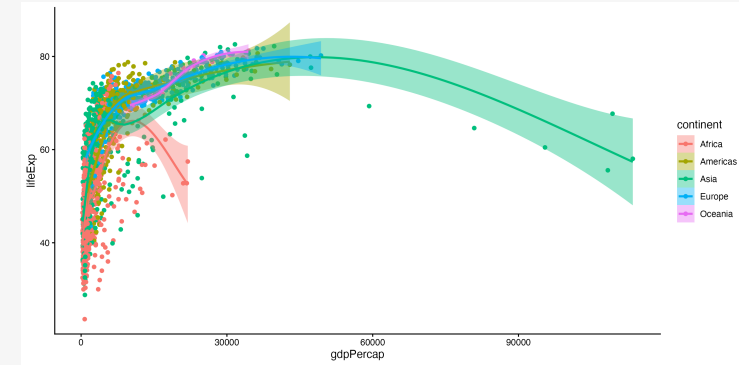
Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y = lifeExp,  
                          color = continent,  
                          fill = continent))  
  
p + geom_point()
```



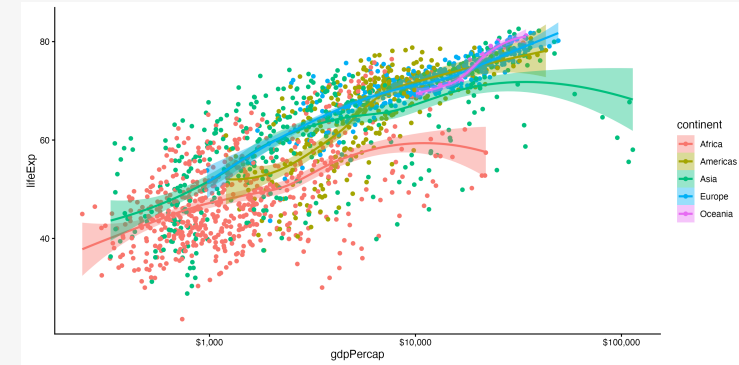
Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y = lifeExp,  
                          color = continent,  
                          fill = continent))  
  
p + geom_point() +  
    geom_smooth(method = "loess")
```



Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y = lifeExp,  
                          color = continent,  
                          fill = continent))  
  
p + geom_point() +  
  geom_smooth(method = "loess") +  
  scale_x_log10(labels = scales::label_dollar())
```

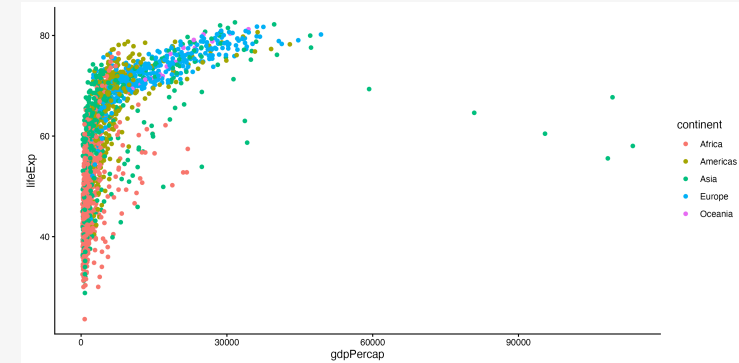


Geoms can take their own mappings

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

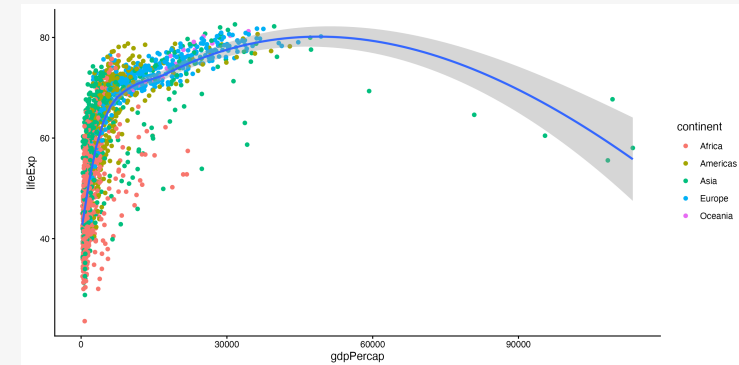
Geoms can take their own mappings

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



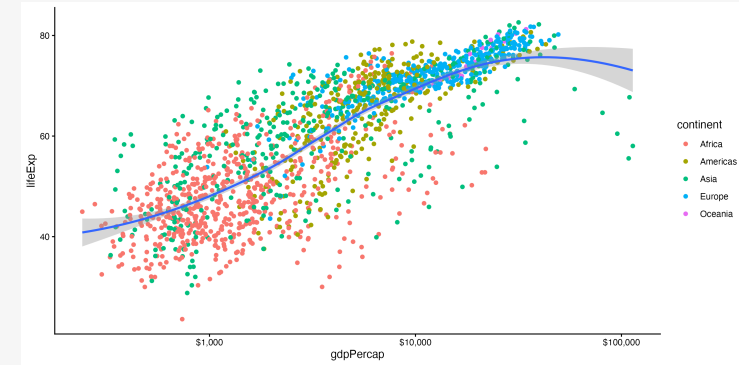
Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
           mapping = aes(x = gdpPercap,  
                         y = lifeExp))  
p + geom_point(mapping = aes(color = continent)) +  
  geom_smooth(method = "loess")
```



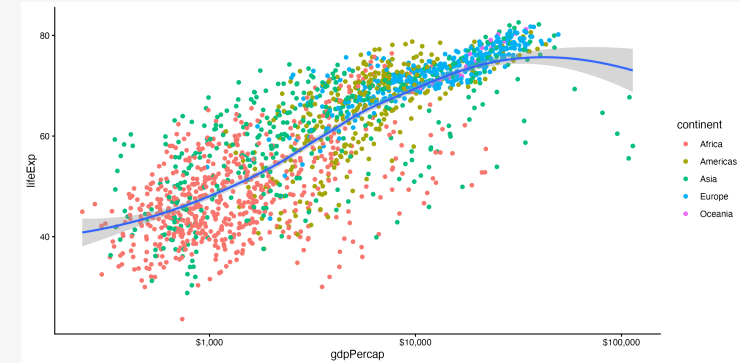
Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y = lifeExp))  
p + geom_point(mapping = aes(color = continent)) +  
  geom_smooth(method = "loess") +  
  scale_x_log10(labels = scales::label_dollar())
```



Geoms can take their own mappings

```
p ← ggplot(data = gapminder,  
            mapping = aes(x = gdpPercap,  
                          y = lifeExp))  
p + geom_point(mapping = aes(color = continent)) +  
  geom_smooth(method = "loess") +  
  scale_x_log10(labels = scales::label_dollar())
```



Pay attention to
which scales and
guides are drawn,
and why

Guides and scales reflect `aes()` mappings

```
mapping = aes(color =  
continent, fill = continent)
```

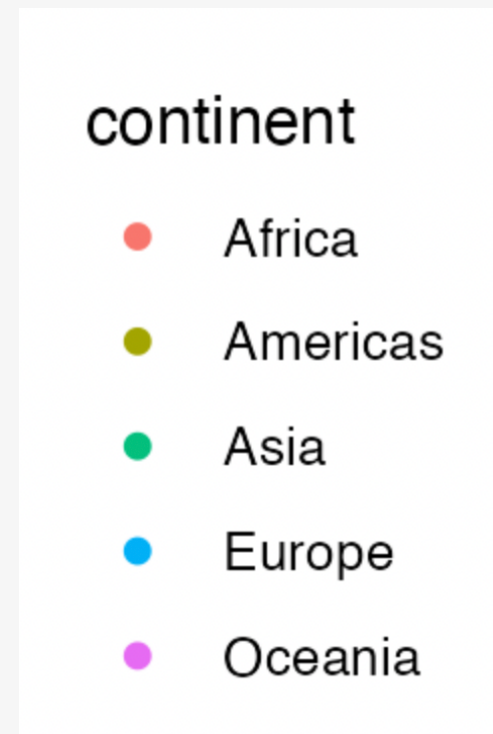


Guides and scales reflect `aes()` mappings

```
mapping = aes(color =  
continent, fill = continent)
```



```
mapping = aes(color =  
continent)
```



**Remember: Every
mapped variable
has a scale**

Saving your work

Use `ggsave()`

```
## Save the most recent plot
ggsave(filename = "figures/my_figure.png")

## Use here() for more robust file paths
ggsave(filename = here("figures", "my_figure.png"))

## A plot object
p_out <- p + geom_point(mapping = aes(color = log(pop))) +
  scale_x_log10()

ggsave(filename = here("figures", "lifexp_vs_gdp_gradient.pdf"),
  plot = p_out)

ggsave(here("figures", "lifexp_vs_gdp_gradient.png"),
  plot = p_out,
  width = 8,
  height = 5)
```

Customize figure output

In single code chunks

Set particular options in any quarto code chunk by including lines like these at the top of the chunk:

```
#!/ fig-height: 8  
#!/ fig-width: 5  
#!/ fig-cap: "A caption"
```

For the whole document

Use the YAML header:

```
format:  
  html:  
    fig-width: 8  
    fig-height: 6  
  pdf:  
    fig-width: 7  
    fig-height: 5
```