

# Annotations

## Session 9

PMAP 8921: Data Visualization with R  
Andrew Young School of Policy Studies  
Fall 2024

# Plan for today

**Fretting the little things**

**Text in plots**

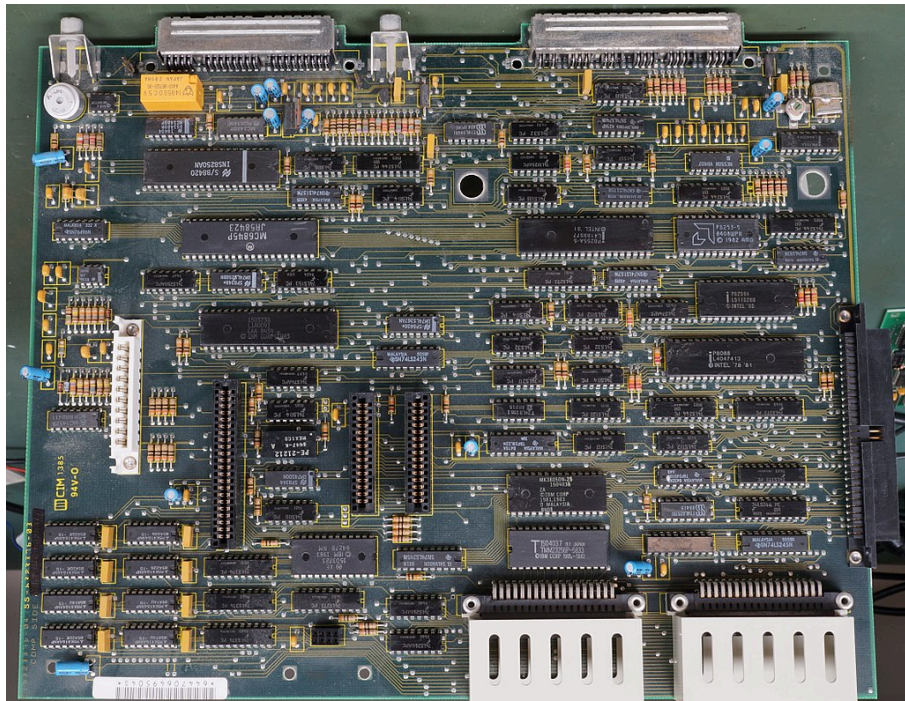
**Seeds**

# Fretting the little things

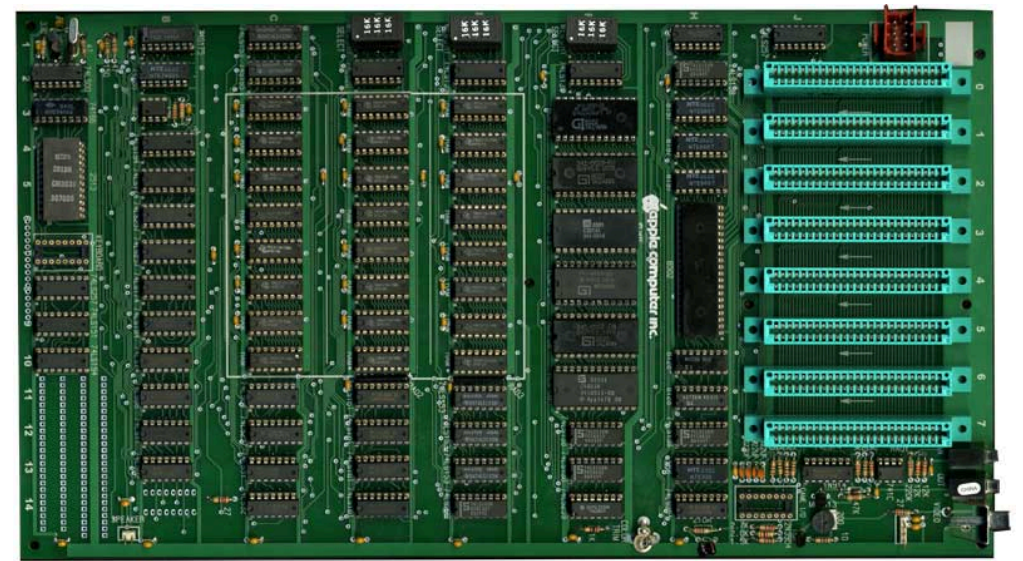
# Little details matter



# Obsession with tiny details



IBM PC Jr.



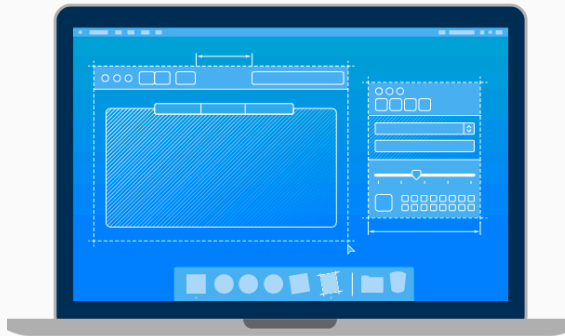
Apple IIe

# Human-focused design

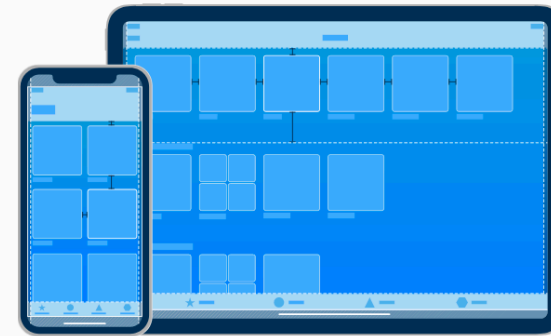
**“This is what customers pay us for—to sweat all these details so it’s easy and pleasant for them to use our computers.”**

# Human Interface Guidelines

Get in-depth information and UI resources for designing great apps that integrate seamlessly with Apple platforms.



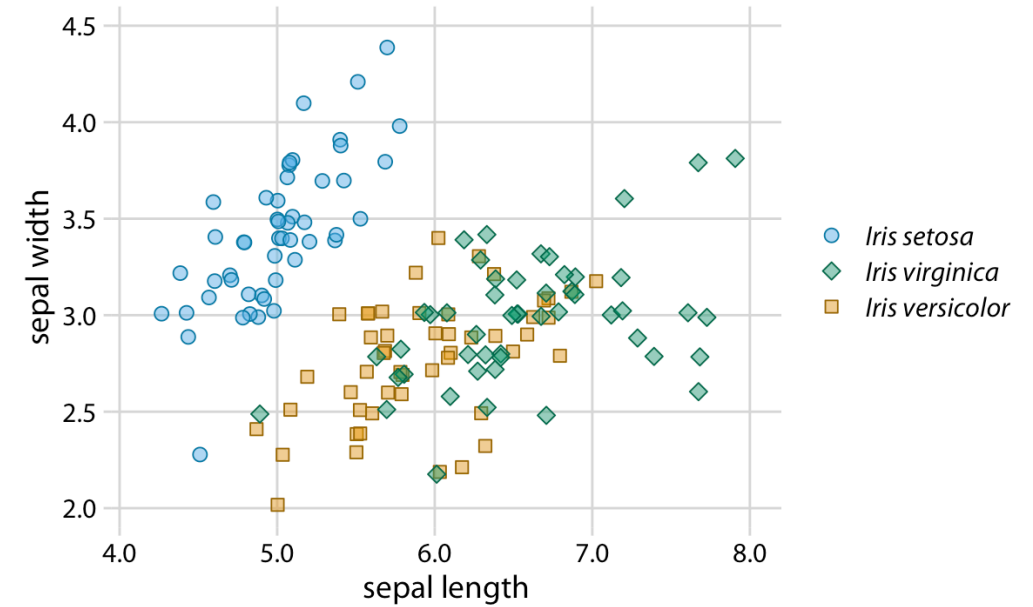
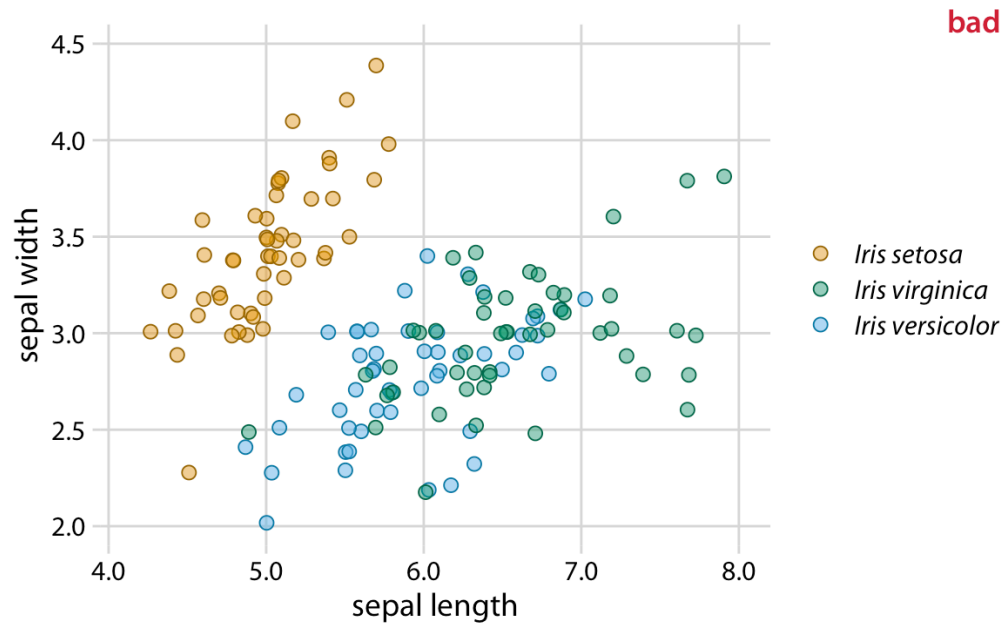
macOS >



iOS >

# Graph details: Redundant coding

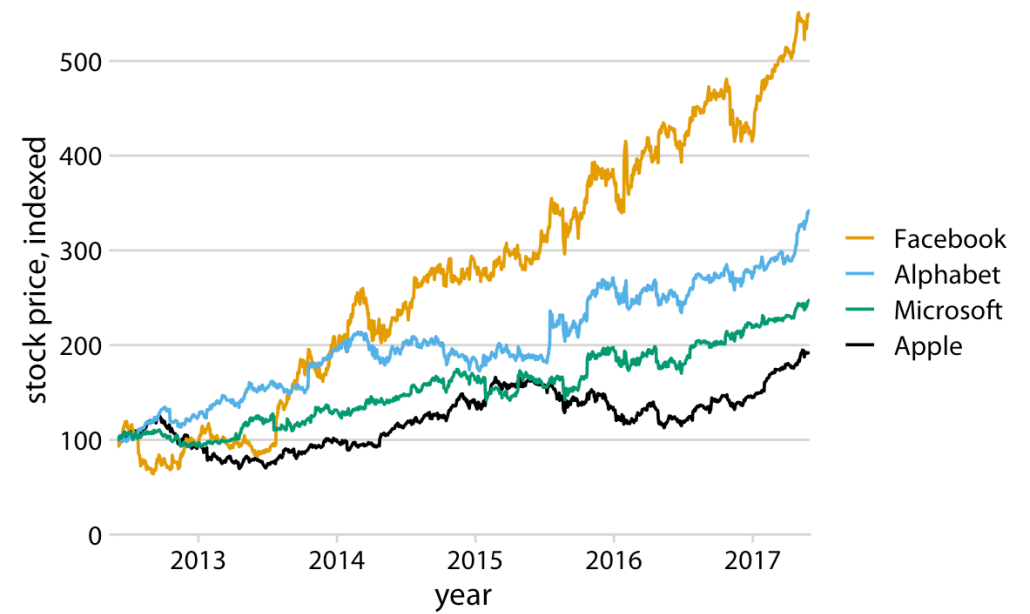
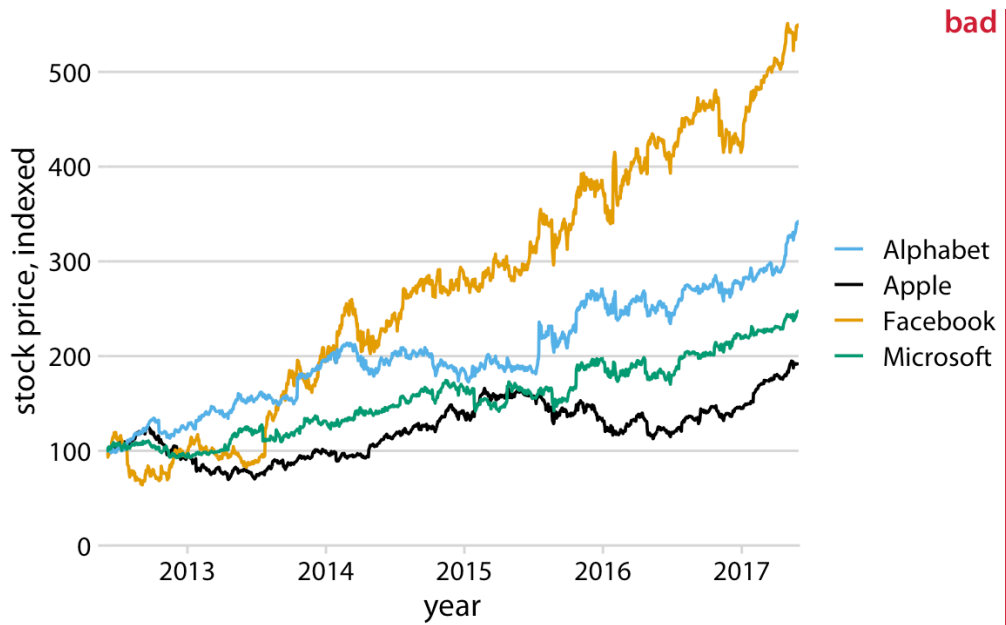
One little change makes this far more accessible





# Graph details: Consistent ordering

Again, one little change makes this far more accessible



# Details matter

**Worrying about tiny details in graphs...**

**...makes them easier for your audience to understand**

**...improves their beauty**

**...enhances the truth**

# Text in plots

# Including text on a plot

**Label actual data points**

`geom_text()`, `geom_label()`, `geom_text_repel()`, etc.

**Add arbitrary annotations**

`annotate()`

**Titles, subtitles, captions, etc.**

`labs(title = "blah", subtitle = "blah", caption = "blah")`

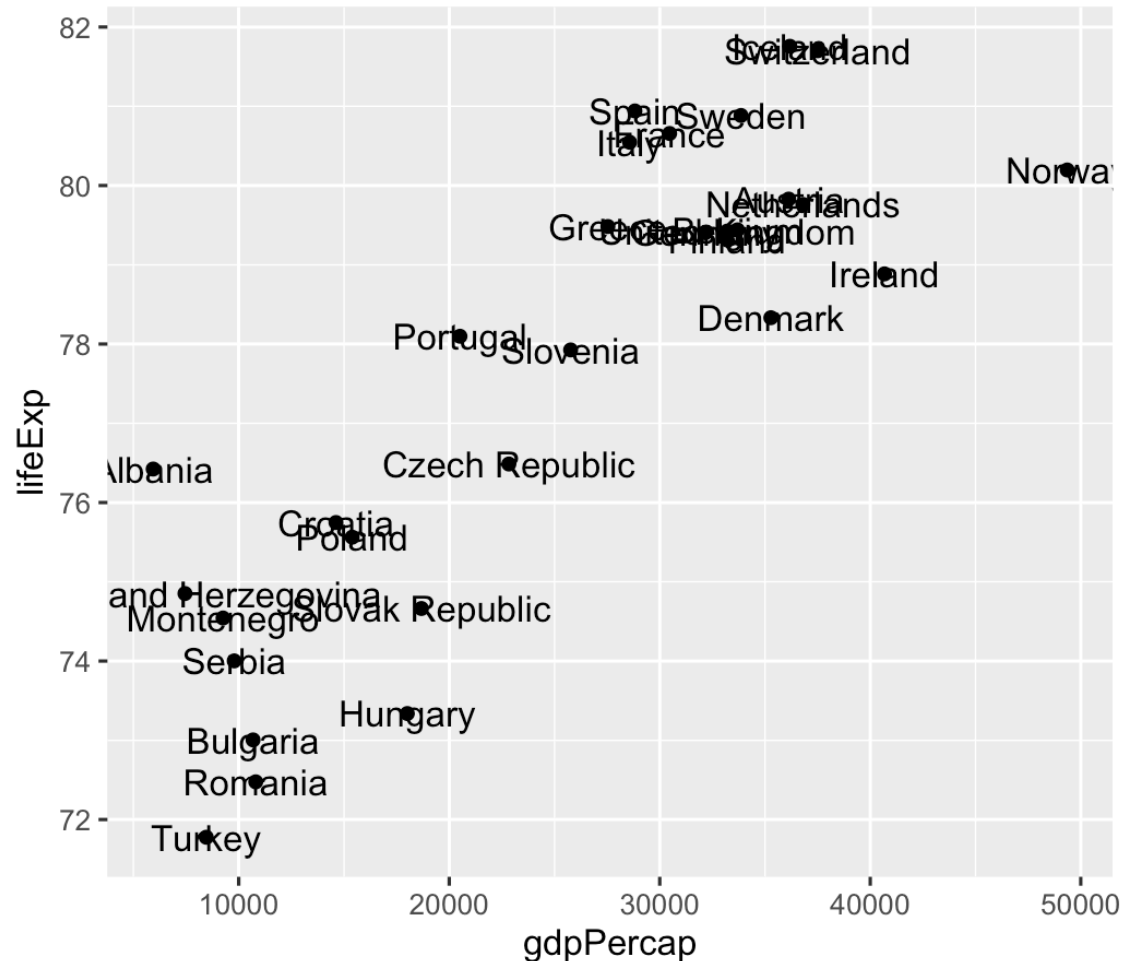
# Label actual data points

```
library(gapminder)

gapminder_europe <- gapminder |>
  filter(year == 2007,
         continent == "Europe")

ggplot(gapminder_europe,
       aes(x = gdpPerCap, y = lifeExp)) +
  geom_point() +
  geom_text(aes(label = country))
```

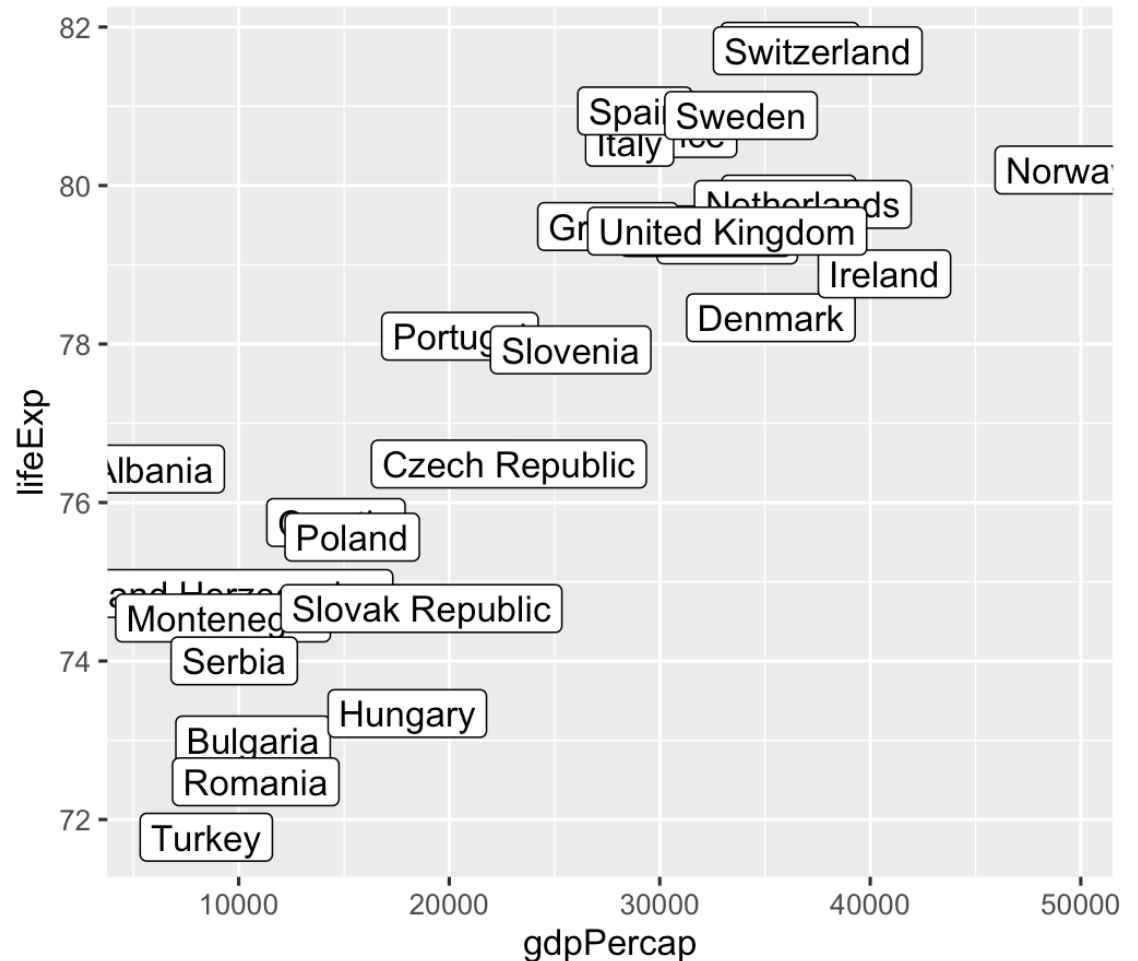
**Ew.**



# Label actual data points

```
ggplot(gapminder_europe,  
       aes(x = gdpPercap, y = lifeExp)) +  
  geom_point() +  
  geom_label(aes(label = country))
```

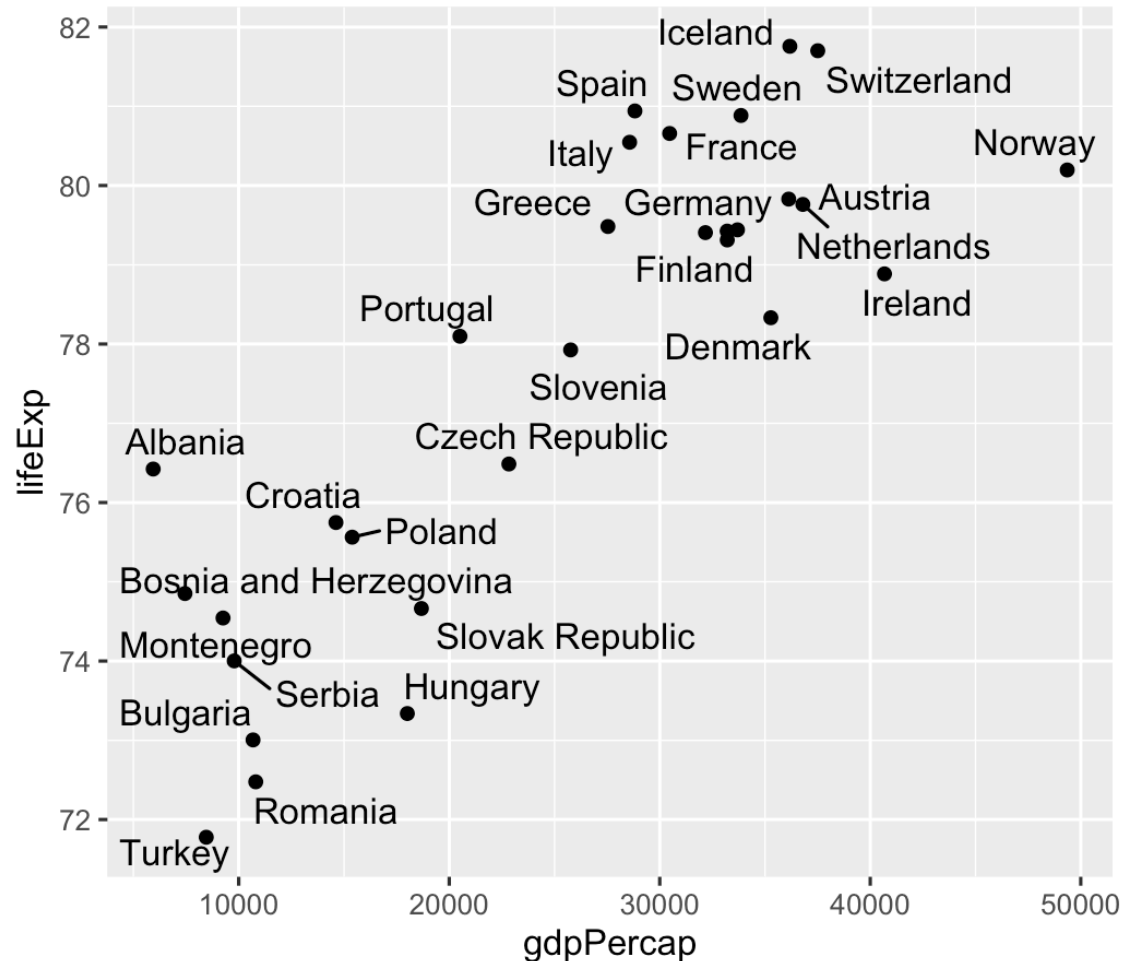
**Still ew. Labels are neat,  
but cover the points.**



# Solution 1: Repel labels

```
library(ggrepel)

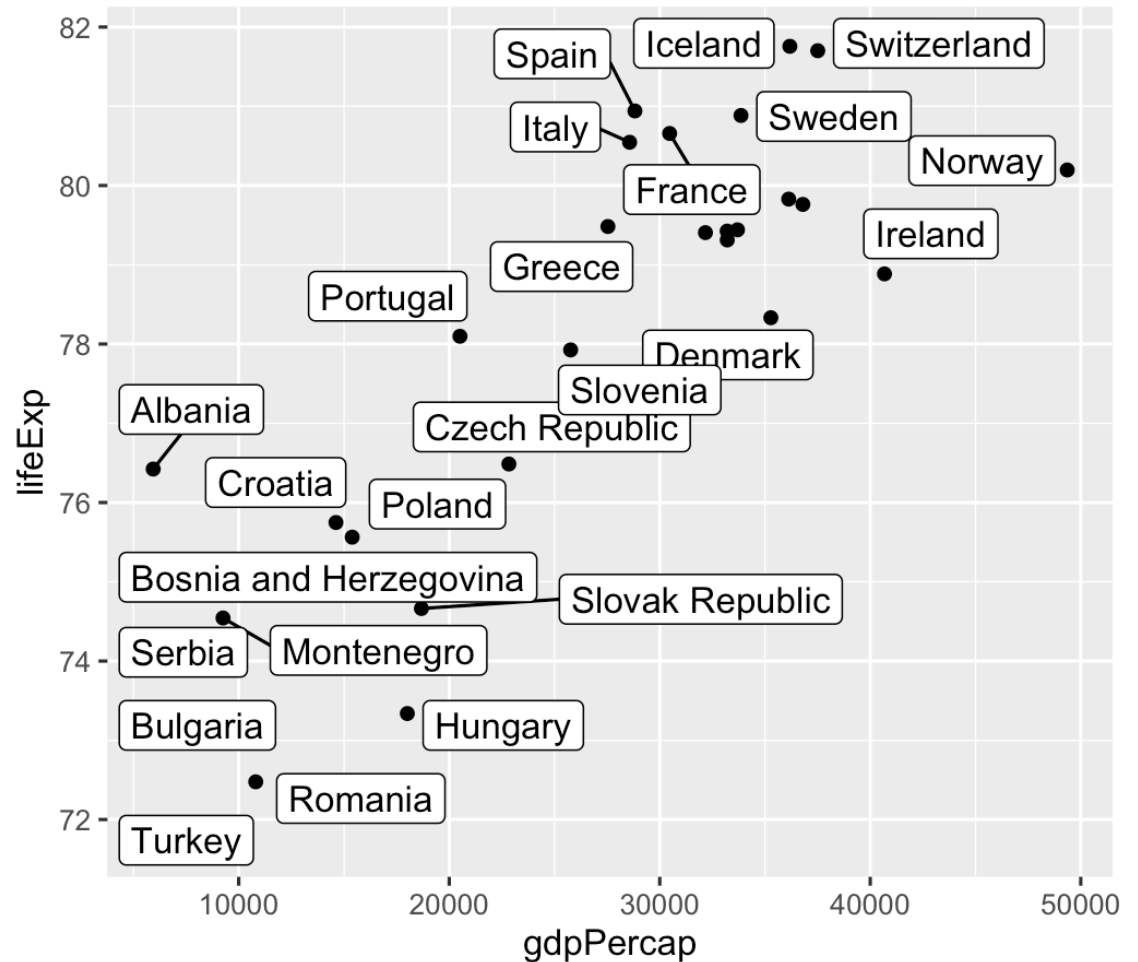
ggplot(gapminder_europe,
       aes(x = gdpPercap, y = lifeExp)) +
  geom_point() +
  geom_text_repel(aes(label = country))
```



# Solution 1: Repel labels

```
library(ggrepel)

ggplot(gapminder_europe,
       aes(x = gdpPercap, y = lifeExp)) +
  geom_point() +
  geom_label_repel(aes(label = country))
```

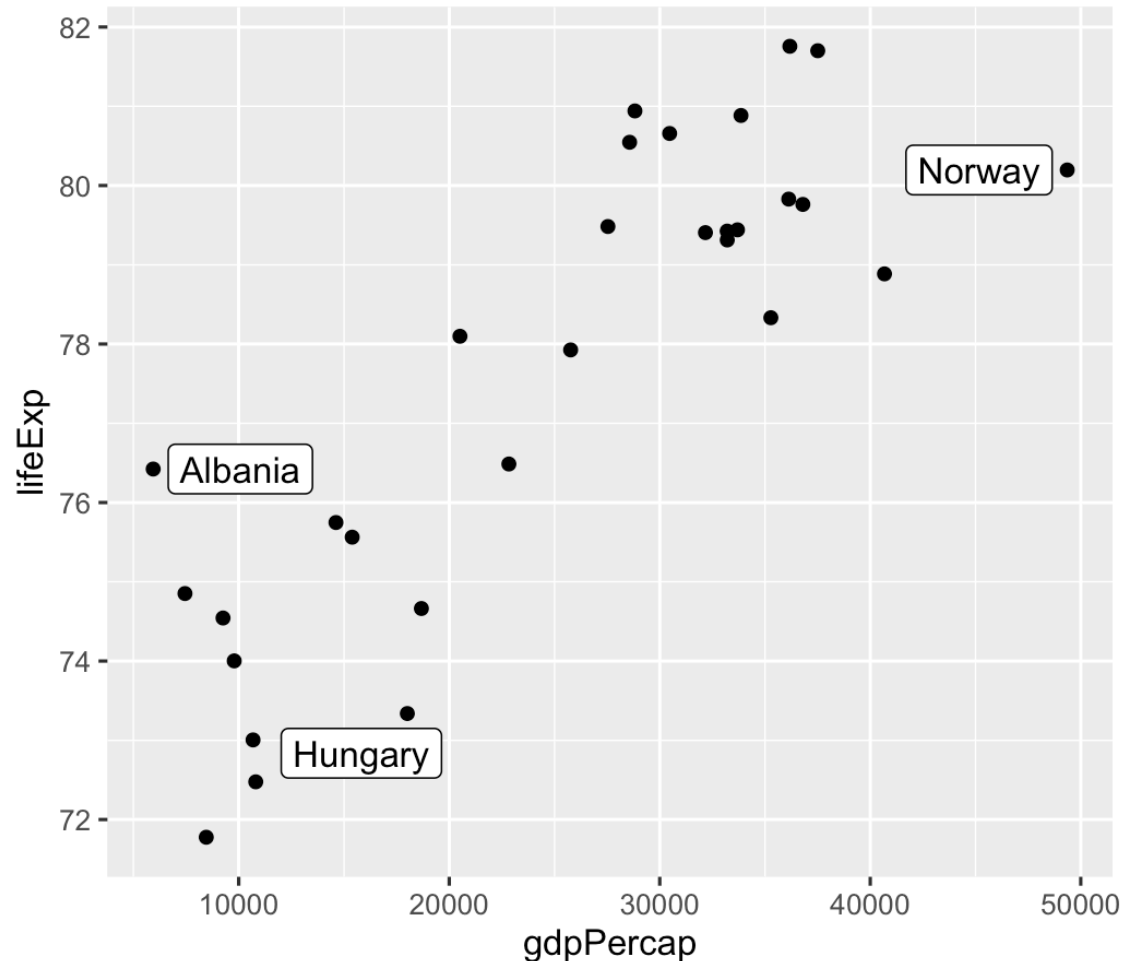




# Solution 2a: Don't use so many labels

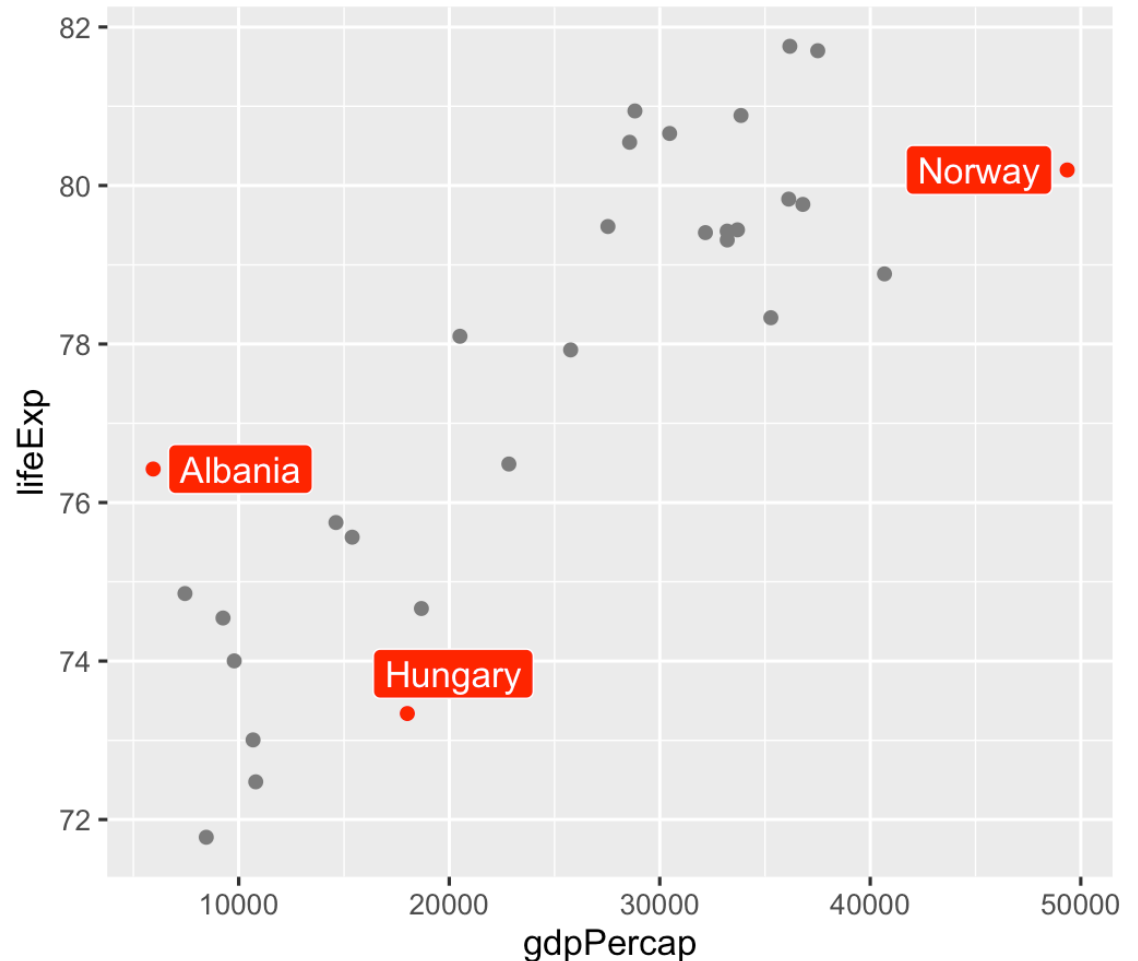
```
gapminder_europe <- gapminder_europe |>
  mutate(should_be_labeled =
    ifelse(country %in% c("Albania",
                          "Norway",
                          "Hungary"),
           TRUE, FALSE))

ggplot(gapminder_europe,
       aes(x = gdpPerCap, y = lifeExp)) +
  geom_point() +
  geom_label_repel(
    data = filter(gapminder_europe,
                  should_be_labeled == TRUE)
    aes(label = country)
  )
```



# Solution 2b: Use other aesthetics too

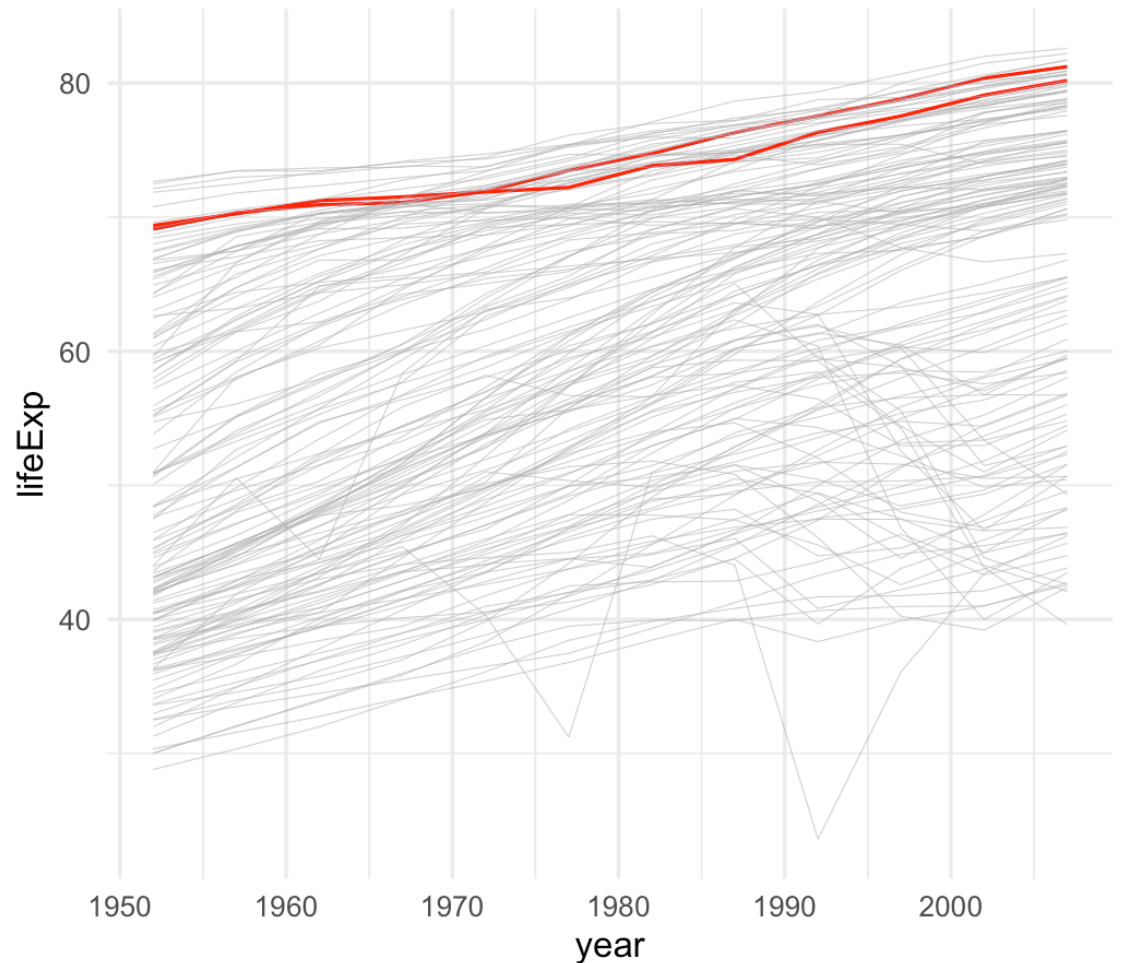
```
ggplot(gapminder_europe,
       aes(x = gdpPercap, y = lifeExp)) +
  geom_point(aes(color = should_be_labeled))
  geom_label_repel(
    data = filter(
      gapminder_europe,
      should_be_labeled == TRUE
    ),
    aes(label = country,
        fill = should_be_labeled),
    color = "white"
  ) +
  scale_color_manual(values = c("grey50",
                                "red")) +
  scale_fill_manual(values = c("red")) +
  guides(color = "none", fill = "none")
```



# (Highlight non-text things too!)

```
# Color just Oceania
gapminder_highlighted <- gapminder |>
  mutate(is_oceania =
    ifelse(continent == "Oceania",
           TRUE, FALSE))

ggplot(gapminder_highlighted,
       aes(x = year, y = lifeExp,
           group = country,
           color = is_oceania,
           linewidth = is_oceania)) +
  geom_line() +
  scale_color_manual(values = c("grey70",
                                "red")) +
  scale_linewidth_manual(values = c(0.1, 0.5)) +
  guides(color = "none", linewidth = "none") +
  theme_minimal()
```



# Including text on a plot

**Label actual data points**

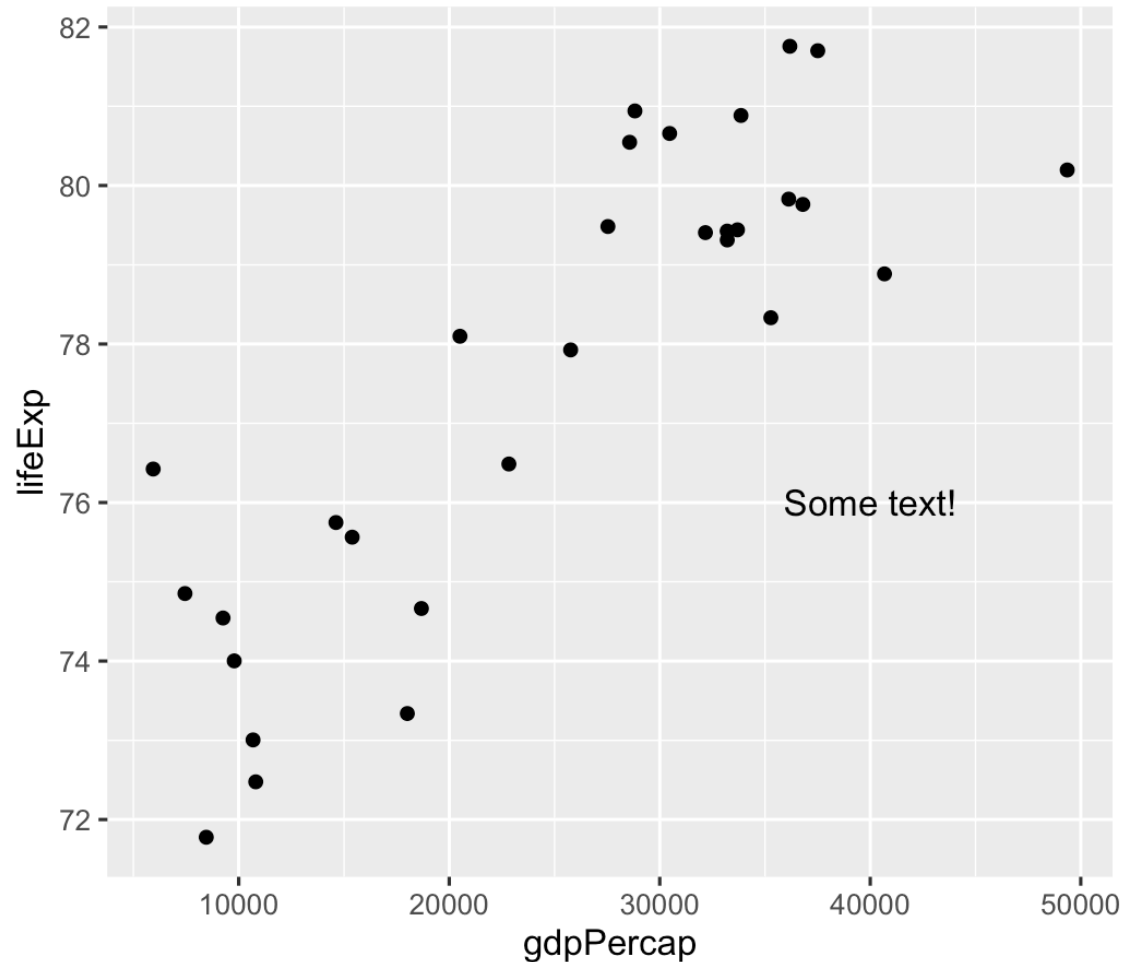
`geom_text()`, `geom_label()`, `geom_text_repel()`, etc.

**Add arbitrary annotations**

`annotate()`

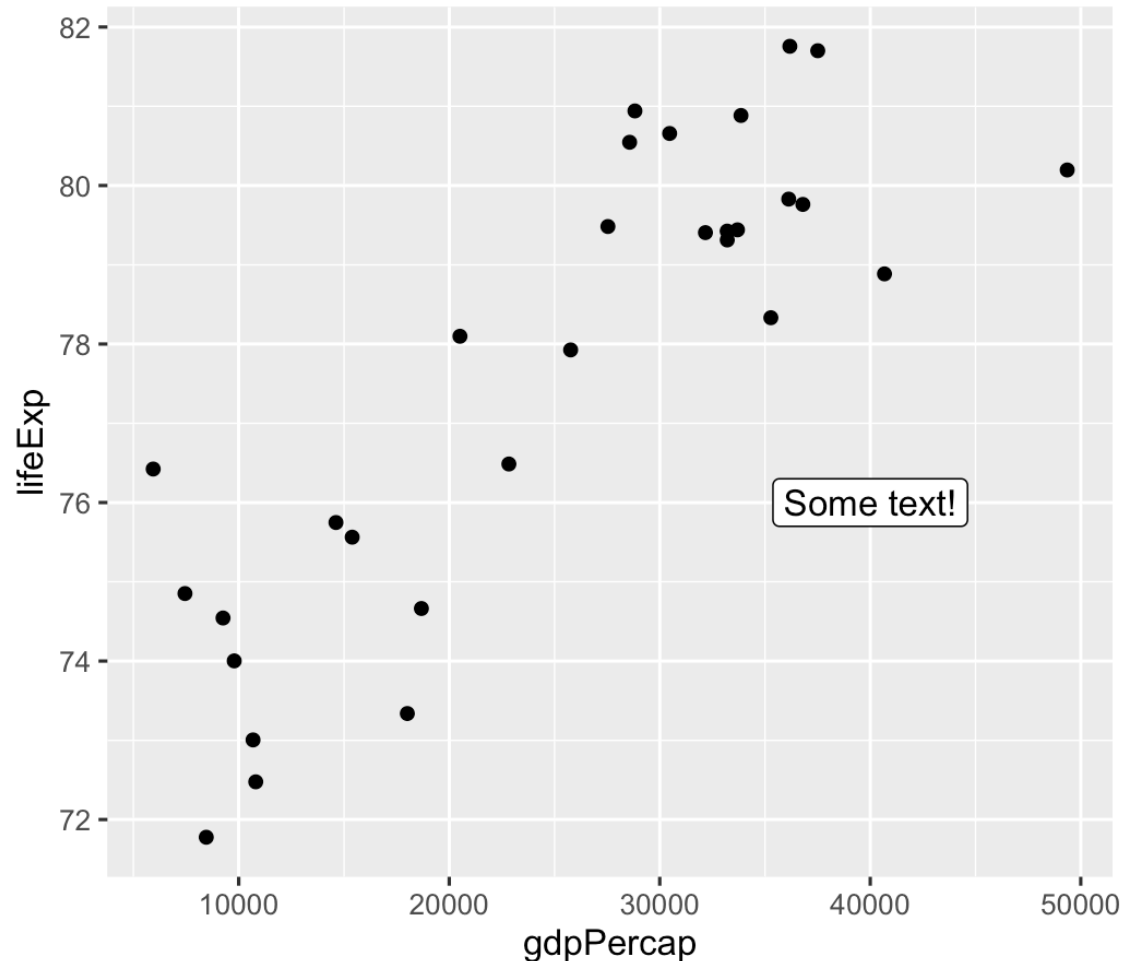
# Adding arbitrary annotations

```
ggplot(gapminder_europe,  
       aes(x = gdpPercap, y = lifeExp)) +  
  geom_point() +  
  annotate(geom = "text",  
         x = 40000, y = 76,  
         label = "Some text!")
```



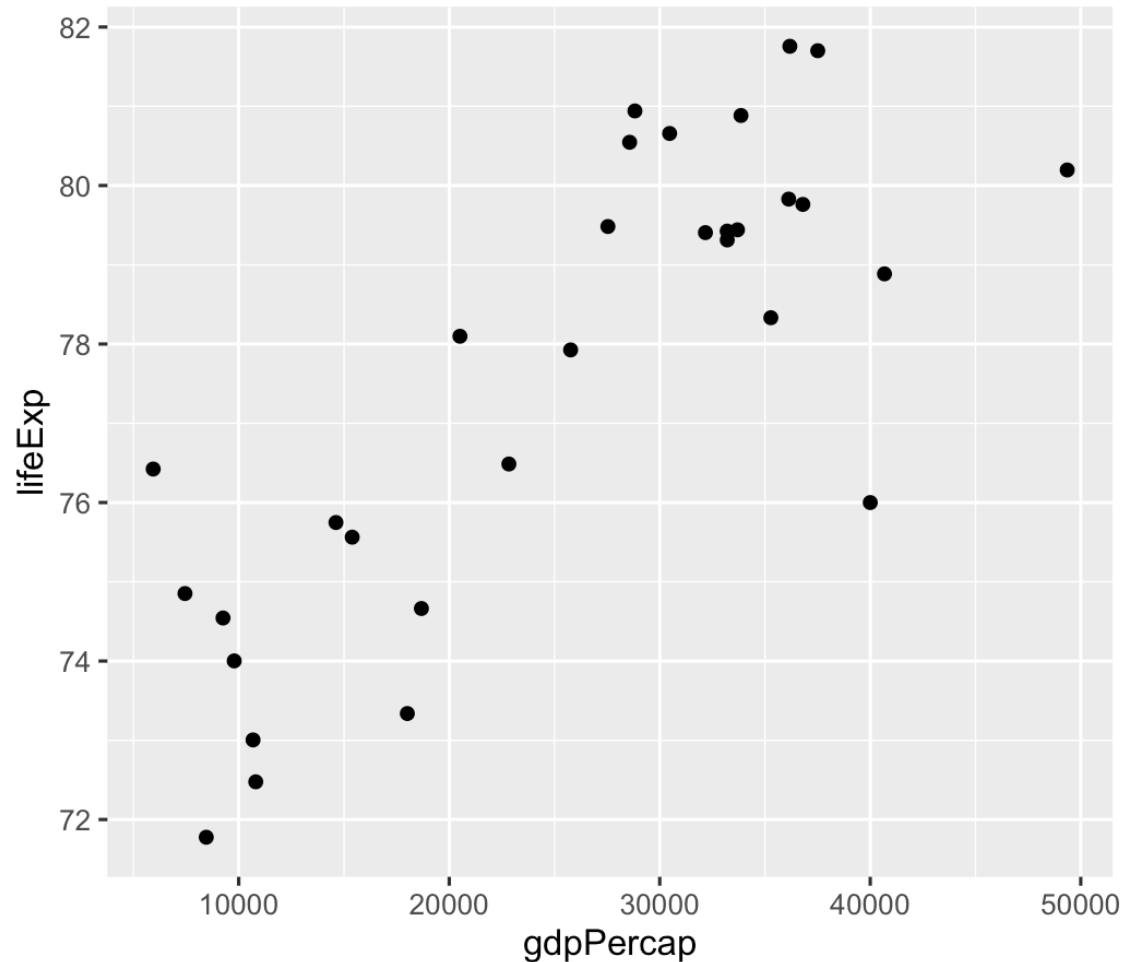
# Adding arbitrary annotations

```
ggplot(gapminder_europe,  
       aes(x = gdpPercap, y = lifeExp)) +  
  geom_point() +  
  annotate(geom = "label",  
         x = 40000, y = 76,  
         label = "Some text!")
```



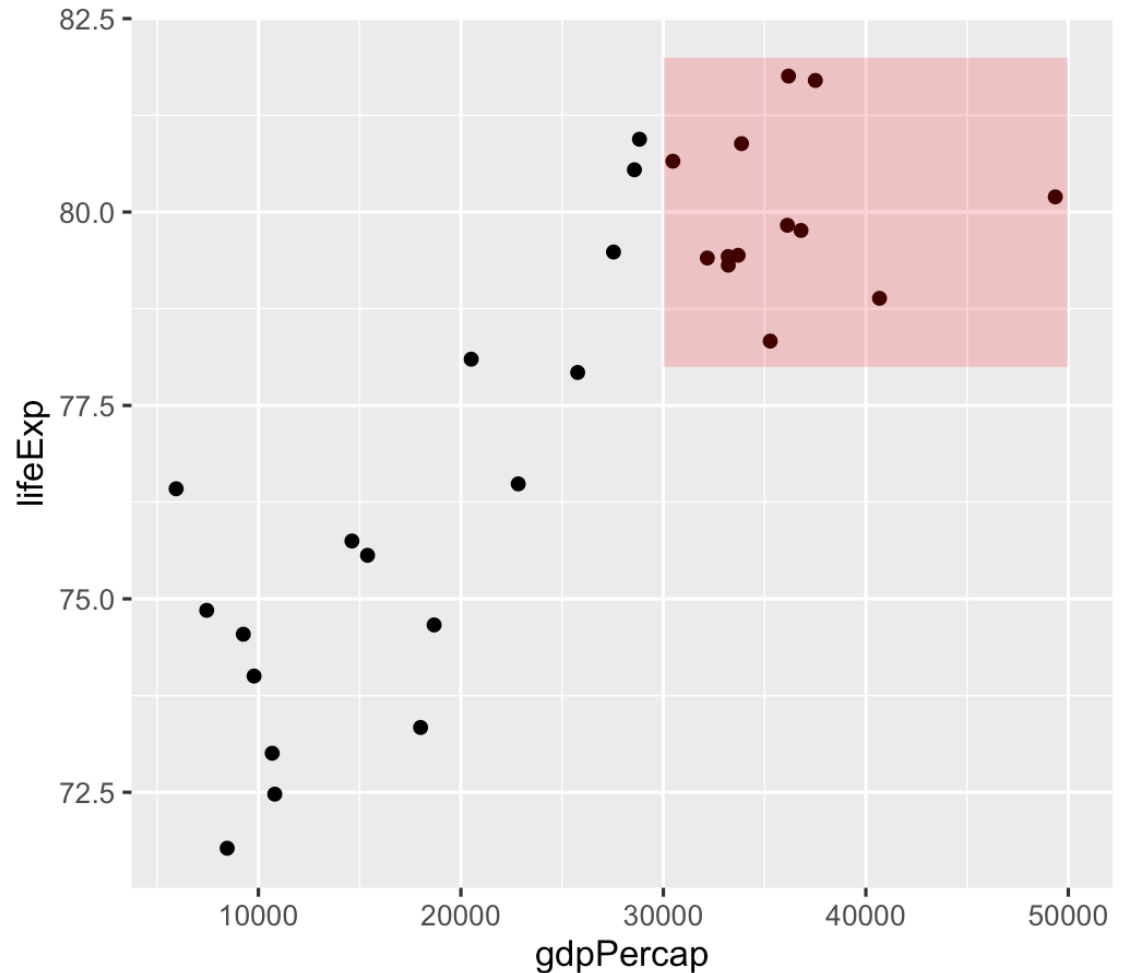
# Any geom works

```
ggplot(gapminder_europe,  
       aes(x = gdpPercap, y = lifeExp)) +  
  geom_point() +  
  # This is evil though!!!  
  # We just invented a point  
  annotate(geom = "point",  
          x = 40000, y = 76)
```



# Any geom works

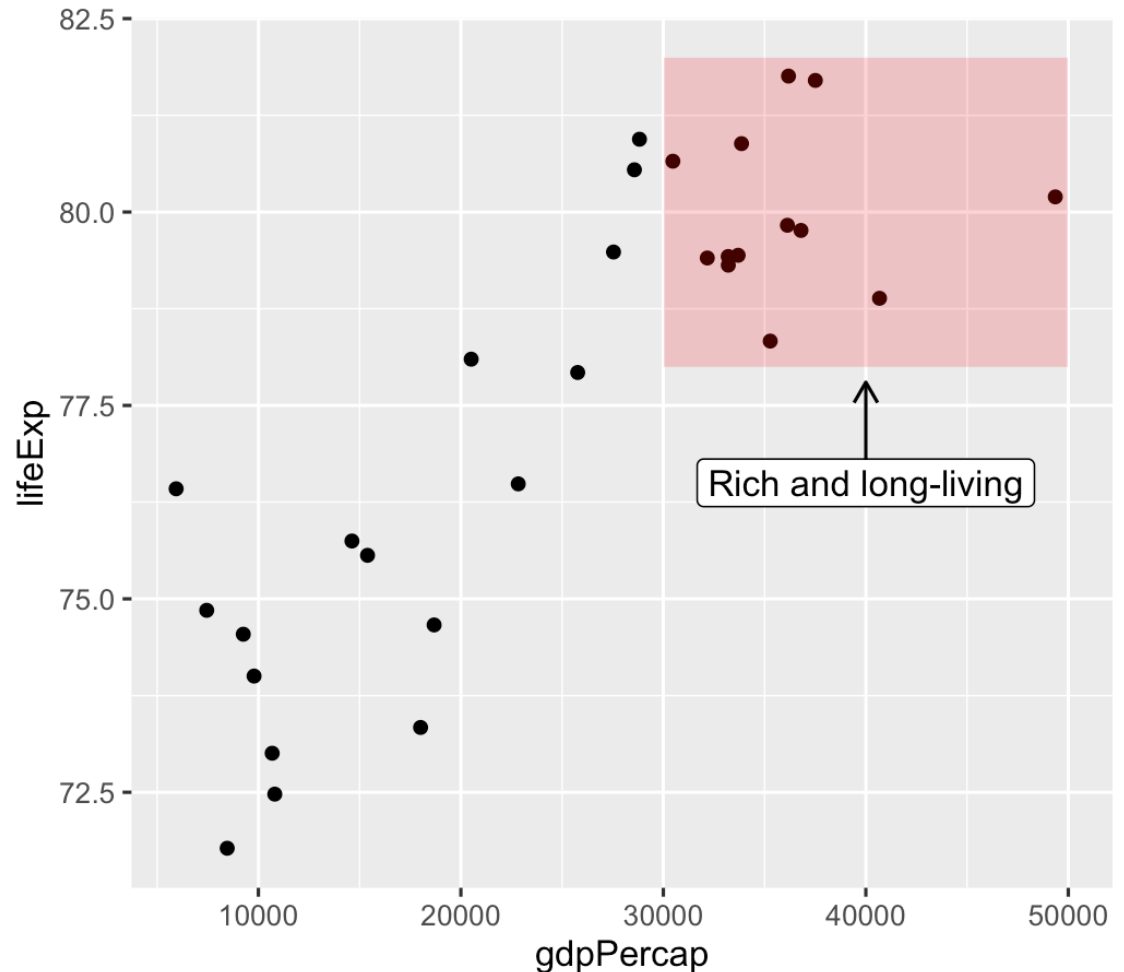
```
ggplot(gapminder_europe,  
       aes(x = gdpPercap, y = lifeExp)) +  
  geom_point() +  
  annotate(geom = "rect",  
         xmin = 30000, xmax = 50000,  
         ymin = 78, ymax = 82,  
         fill = "red", alpha = 0.2)
```





# Use multiple annotations

```
ggplot(gapminder_europe,
       aes(x = gdpPerCap, y = lifeExp)) +
  geom_point() +
  annotate(geom = "rect",
         xmin = 30000, xmax = 50000,
         ymin = 78, ymax = 82,
         fill = "red", alpha = 0.2) +
  annotate(geom = "label",
         x = 40000, y = 76.5,
         label = "Rich and long-living") +
  annotate(geom = "segment",
         x = 40000, xend = 40000,
         y = 76.8, yend = 77.8,
         arrow = arrow(
           length = unit(0.1, "in")))
```



# Including text on a plot

**Label actual data points**

`geom_text()`, `geom_label()`, `geom_text_repel()`, etc.

**Add arbitrary annotations**

`annotate()`

**Titles, subtitles, captions, etc.**

`labs(title = "blah", subtitle = "blah", caption = "blah")`

# Which is better?

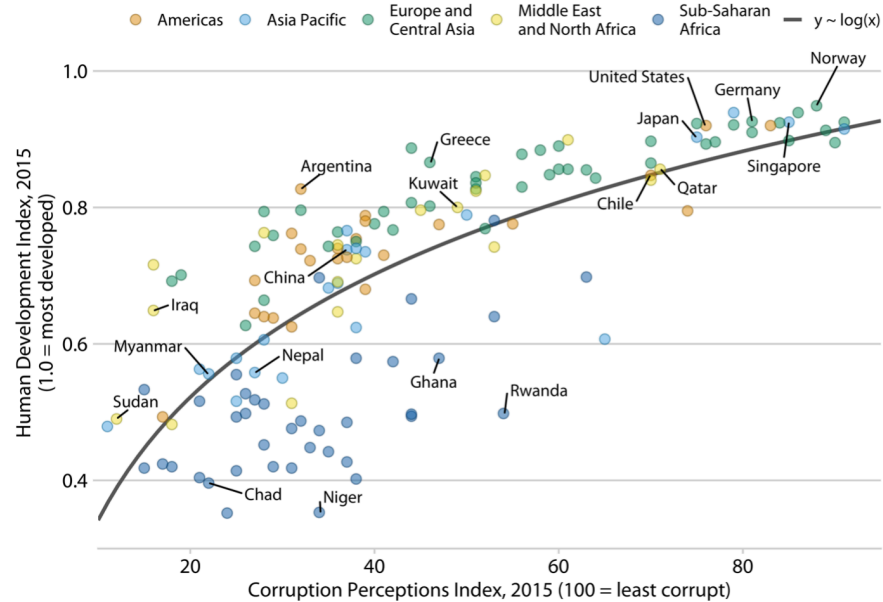
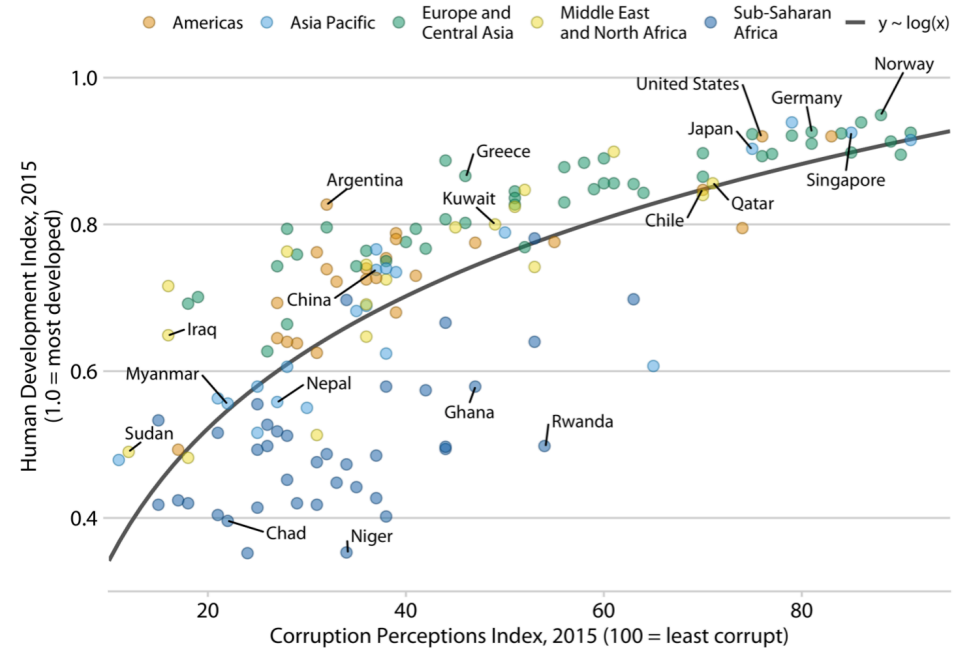


Figure 22.1: Corruption and human development: The most developed countries experience the least corruption. This figure was inspired by a posting in The Economist online (2011). Data sources: Transparency International & UN Human Development Report

## Corruption and human development

The most developed countries experience the least corruption



Data sources: Transparency International & UN Human Development Report

**Neither! Depends on the final document output.**

# Seeds

# Pseudorandomness

**Your computer uses a complicated algorithm to generate random numbers**

**Different programs use different algorithms**

**You can actually sometimes reverse engineer the algorithm!**



**These algorithms all start with something called a "seed", or some number**

**In R this is the current time on your computer + the internal program process ID**

**If two random functions use the same seed, they'll create the same numbers**

# Seeds

Open R on your computer and run this:

```
rnorm(3)
```

You'll generate 3 random numbers from a normal distribution with a mean of 0 and a standard deviation of 1.

They will 100% **not** be these 3 numbers:

```
-1.033, -0.949, and 1.394
```

# Seeds

Now run these two lines in R:

```
set.seed(1234)
```

```
rnorm(3)
```

You'll again generate 3 random numbers,  
but they will **100%** be these:

```
-1.207, 0.277, and 1.084
```



# Why should we care?

Because we set a seed the random numbers will be the same random numbers every time

Reproducible simulations

Reproducible Bayesian models

Jittering in plots

`geom_text_repel()` in plots

# What is a good seed?

Any whole number

1234(567)

1

13, 42, 8675309, or your favorite number

20200519

[Random.org](https://www.random.org) atmospheric noise

# Best practice

If you're doing *anything* with randomness, include `set.seed(SOME_NUMBER)` at the beginning of your document

Some functions have a `seed` argument—use it

```
geom_label_repel(..., seed = 1234)
```

```
position_jitter(..., seed = 1234)
```

# Example

```
ggplot(mpg, aes(x = drv, y = hwy)) +  
  geom_point(position =  
    position_jitter(seed = 1234,  
                    width = 0.3))
```

**As long as the seed is 1234,  
those dots will always  
be in those exact spots  
on any computer running R**

