

Data Visualization Using R & ggplot2

Naupaka Zimmerman (@naupakaz), Andrew Tredennick (@ATredennick),
and Karthik Ram (@_inundata)

August 6, 2017

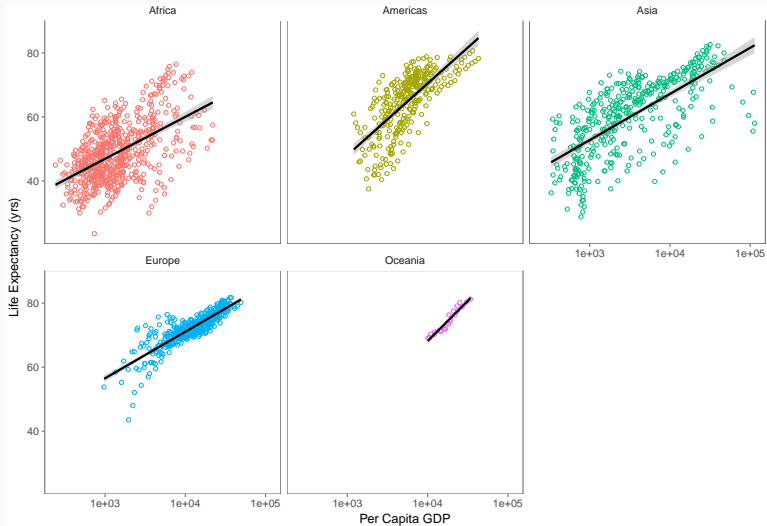
Some housekeeping

Install some packages

```
install.packages("ggplot2", dependencies = TRUE)  
install.packages("ggthemes")  
install.packages("tidyr")  
install.packages("dplyr")
```

Why ggplot?

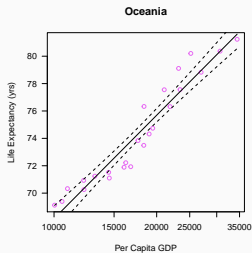
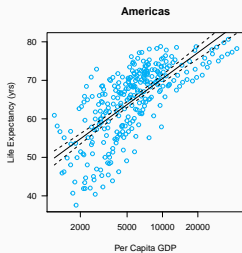
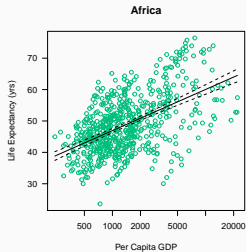
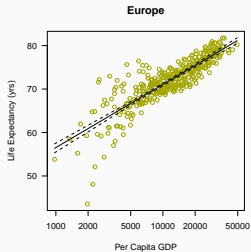
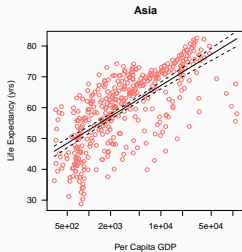
1 minute



1 minute

```
library(ggplot2)
library(gapminder)
library(ggthemes)

ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point(shape = 1, aes(color = continent)) +
  stat_smooth(method = "lm", size = 1, color = "black") +
  scale_x_log10() +
  xlab("Per Capita GDP") +
  ylab("Life Expectancy (yrs)") +
  facet_wrap(~continent) +
  theme_few() +
  guides(color = FALSE)
```



```
library(scales)
library(gapminder)
gapminder <- as.data.frame(gapminder)
conts <- unique(gapminder[, "continent"])
cols <- scales::hue_pal()(length(conts))
par(mfrow = c(2,3))
counter <- 1
for (i in conts) {
  plot(gapminder[which(gapminder$continent == i), "gdpPerCapita"],
       gapminder[which(gapminder$continent == i), "lifeExp"],
       xlab = "Per Capita GDP", ylab = "Life Expectancy (years)",
       main = i, las = 1, log = "x")
  fit <- lm(gapminder[which(gapminder$continent == i), "lifeExp"] ~
```

But wait, there's more...

```
log(gapminder[which(gapminder$continent == i), "gdpPercap"]  
  pred <- predict(fit, interval = "confidence")  
  lines(sort(gapminder[which(gapminder$continent == i), "gdpPercap"]  
    lines(sort(gapminder[which(gapminder$continent == i), "gdpPercap"]  
      lines(sort(gapminder[which(gapminder$continent == i), "gdpPercap"]  
        counter <- counter + 1  
    }
```


Why ggplot?

Why ggplot?

- More elegant and compact code than with base graphics
- More aesthetically pleasing defaults than lattice
- Very powerful for exploratory data analysis

Why ggplot?

- `gg` is for **grammar of graphics** (term by Lee Wilkinson)
- A set of terms that defines the basic components of a plot
- Used to produce figures using coherent, consistent syntax

Why ggplot2?

- Supports a continuum of expertise
- Easy to get started, plenty of power for complex figures

The Grammar

Data

- Must be a data frame (`data.frame()`, `as.data.frame()`)
- Gets pulled into the `ggplot()` object

A quick example

The iris dataset

```
head(iris)
```

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 1	5.1	3.5	1.4	0.2	setosa
## 2	4.9	3.0	1.4	0.2	setosa
## 3	4.7	3.2	1.3	0.2	setosa
## 4	4.6	3.1	1.5	0.2	setosa
## 5	5.0	3.6	1.4	0.2	setosa
## 6	5.4	3.9	1.7	0.4	setosa

Helps your data play nice with ggplot


```
iris[1:2, ]
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1         5.1         3.5         1.4         0.2   setosa
## 2         4.9         3.0         1.4         0.2   setosa
```

```
library(tidyr)
```

```
df <- gather(iris, key = flower_attribute,
              value = measurement, -Species)
```

```
df[1:2, ]
```

```
##   Species flower_attribute measurement
## 1  setosa   Sepal.Length         5.1
## 2  setosa   Sepal.Length         4.9
```

Aesthetics

aesthetics

- How your data are represented visually
- a.k.a. mapping
- which data on the x
- which data on the y
- but also: **color**, **size**, shape, transparency

An example

```
myplot <- ggplot(data = iris, aes(x = Sepal.Length,  
                                  y = Sepal.Width))
```

```
summary(myplot)
```

```
## data: Sepal.Length, Sepal.Width, Petal.Length, Petal.Width
```

```
## Species [150x5]
```

```
## mapping: x = Sepal.Length, y = Sepal.Width
```

```
## faceting: <ggproto object: Class FacetNull, Facet>
```

```
## compute_layout: function
```

```
## draw_back: function
```

```
## draw_front: function
```

```
## draw_labels: function
```

```
## draw_panels: function
```

```
## finish_data: function
```

```
## init_scales: function
```