Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Data Wrangling and Visualisation Using R

Stefan Müller

4 December 2017

http://muellerstefan.net/data-workshop

Install and load software and packages

Install packages

We will work with the `dplyr` and `ggplot2` packages. `tidyverse` contains a collection of useful packages.

```
install.packages("dplyr")
install.packages("ggplot2")
```

Having installed the package, we need to load them in the R environment.

```
library(dplyr)
library(ggplot2)
```

Note: `install.packages("tidyverse")` loads all packages from the tidyverse.

Gapminder: our sample dataset

We will use the `gapminder` dataset throughout this course.

```
install.packages("gapminder")
```

```
library(gapminder)
```

Getting to know a dataset

Understand the structure and variable coding of a dataset

```
str(gapminder)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1704 obs. of  6
##  $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1
##  $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3
##  $ year     : int  1952 1957 1962 1967 1972 1977 1982 1987 19
##  $ lifeExp  : num  28.8 30.3 32 34 36.1 ...
##  $ pop      : int  8425333 9240934 10267083 11537966 13079460
##  $ gdpPercap: num  779 821 853 836 740 ...
```

Print the first five rows

```
head(gapminder)
```

country	continent	year	lifeExp	pop	gdpPercap
Afghanistan	Asia	1952	28.801	8425333	779.4453
Afghanistan	Asia	1957	30.332	9240934	820.8530
Afghanistan	Asia	1962	31.997	10267083	853.1007
Afghanistan	Asia	1967	34.020	11537966	836.1971
Afghanistan	Asia	1972	36.088	13079460	739.9811
Afghanistan	Asia	1977	38.438	14880372	786.1134

We can find out more information on the data using `?gapminder`.

Getting started with data wrangling

Create a new object

```
one <- 1
two <- 2

one + two
```

```
## [1] 3
```

```
one / two
```

```
## [1] 0.5
```

What you will learn tody

- Data wrangling with `dplyr`
  - `select`: pick columns by name
  - `filter`: keep rows matching criteria
  - `arrange`: reorder rows
  - `mutate`: add new variables
  - `group_by`: group rows by category
  - `summarise`: reduce variables to values
- Data visualisation with `ggplot2`

Data wrangling: Sample data frame

```
(df <- data.frame(
  colour = c("green", "black", "black",
             "green", "black", "red"),
  value = 1:6))
```

colour	value
green	1
black	2
black	3
green	4
black	5
red	6

`filter()`

Filter a data frame

```
filter(df, colour == "green")
```

colour	value
green	1
green	4

```r
filter(df, colour != "green")
```

colour	value
black	2
black	3
black	5
red	6

```r
filter(df, colour %in% c("green", "red"))
```

colour	value
green	1
green	4
red	6

```r
filter(df, value >= 5)
```

colour	value
black	5
red	6

The pipe (%>%) operator

Hint: pronounce %>% as *then*.

```
df %>%
  filter(colour != "red") %>%
  filter(value >= 5)
```

colour	value
black	5

```
df %>%
  filter(colour != "red" & value >= 5)
```

colour	value
black	5

Exercise

1. Load `gapminder` dataset (`library(gapminder)`.
2. Type `?gapminder` and `View(gapminder)` to inspect the data.
3. Filter only observations from Ireland (call new object `gap_ire`)
4. Filter only observations from Europe or Africa (`gap_europe_africa`)

Solution

```
# load gapminder data
library(gapminder)

# variable names
names(gapminder)
```

```
## [1] "country"   "continent" "year"      "lifeExp"   "pop"
```

```
# number of rows and columns
dim(gapminder)
```

```
## [1] 1704    6
```

Solution

```
gap_ire <- gapminder %>%
  filter(country == "Ireland")

nrow(gap_ire)
```

```
## [1] 12
```

```
gap_europe_africa <- gapminder %>%
  filter(continent %in% c("Europe", "Africa"))
```

Data visualisation with `ggplot2`

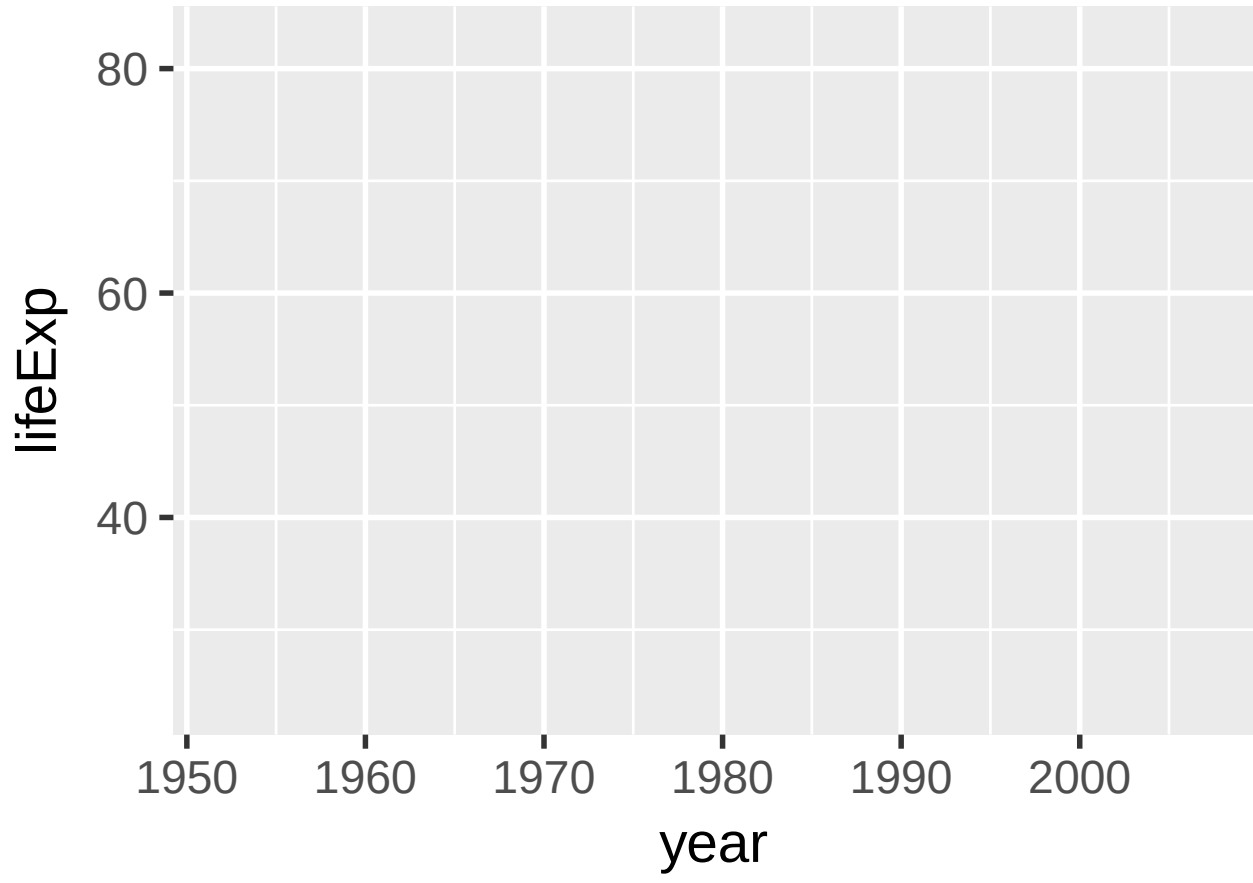
The ggplot2 package

Basic structure:

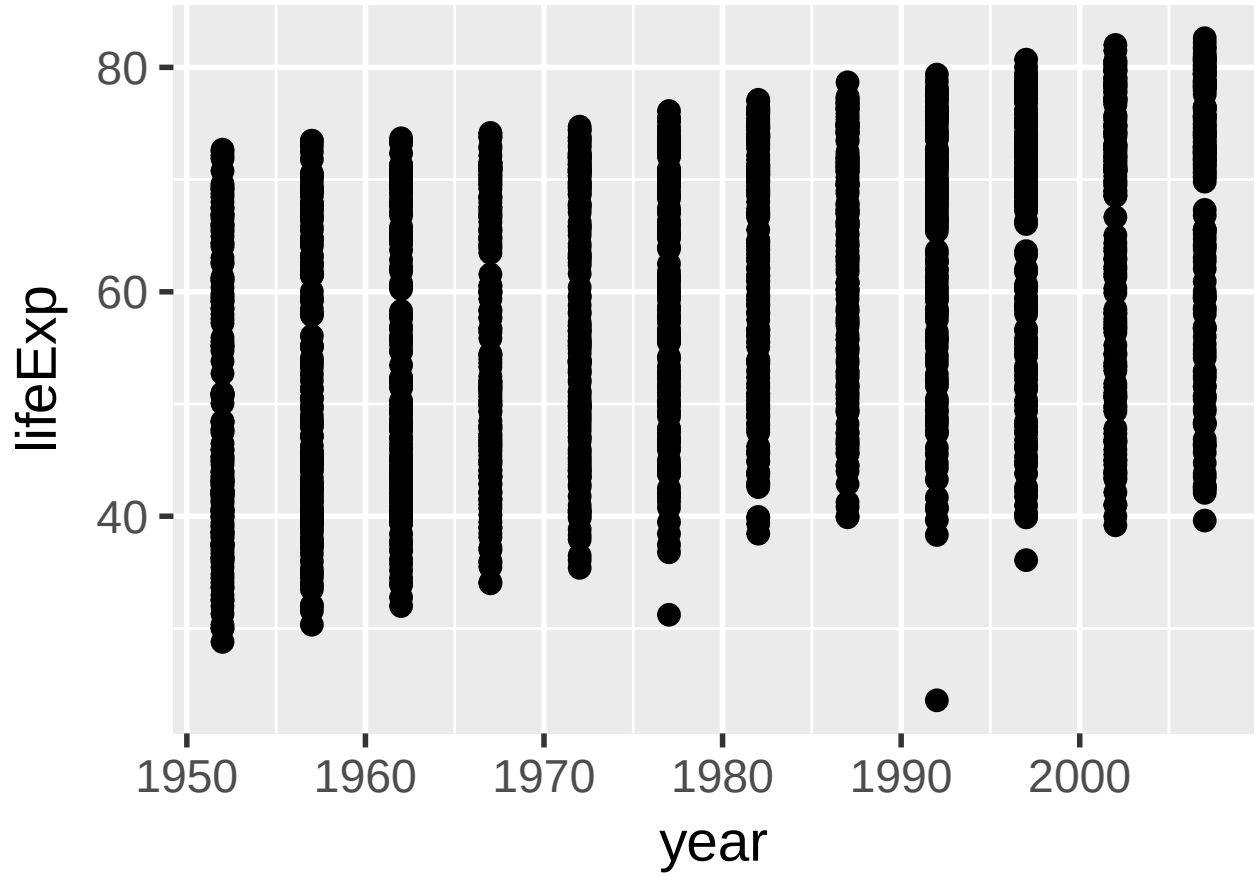
1. call `ggplot()`
2. specify data with `data`
3. specify coordinate system/axes with `aes()`

```
plot1 <- ggplot(data = gapminder,
                aes(x = year, y = lifeExp))
```

```
plot1
```



```
plot1 +
  geom_point()
```



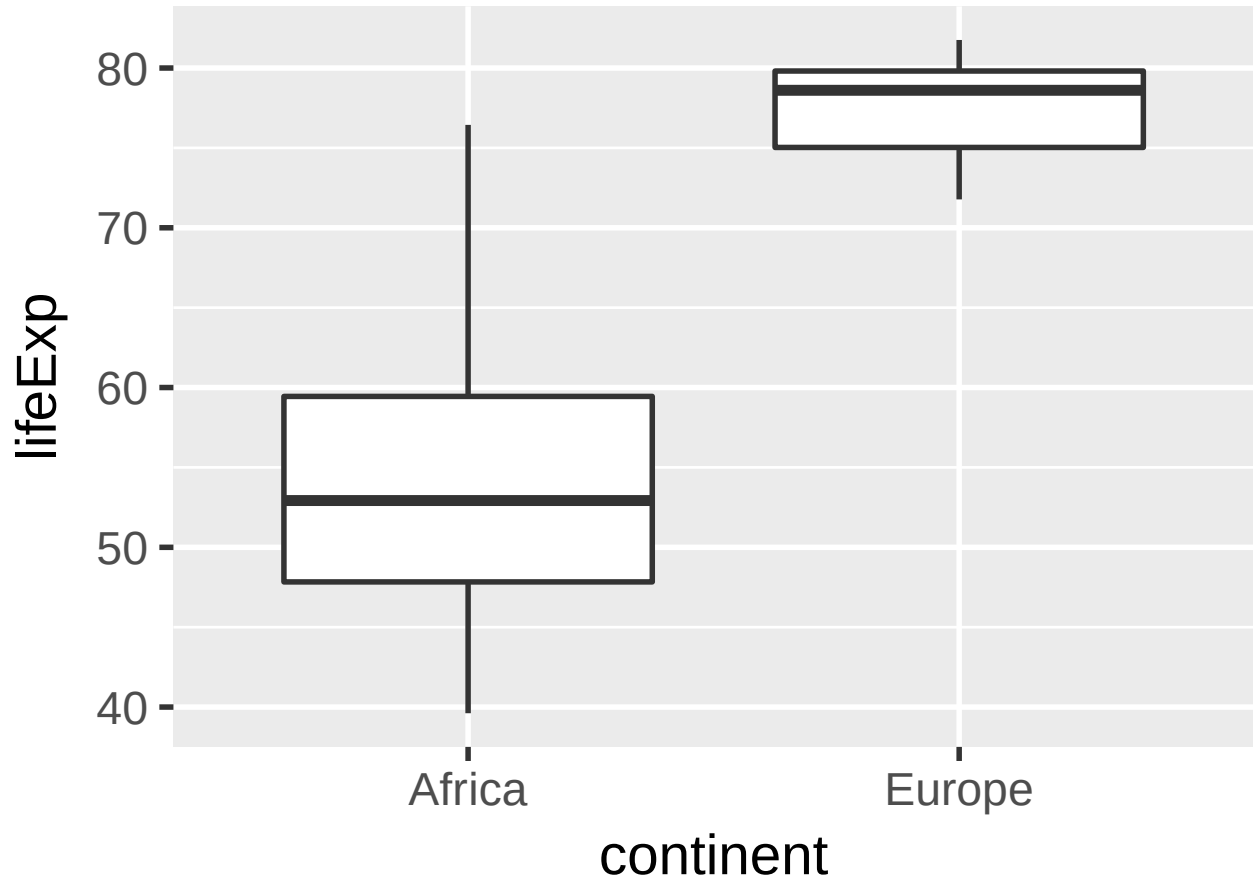
Hands-on Exercise

1. Use `gapminder`, create data frame with only with observations from Europe or Africa in the year 2007
2. Create a boxplot with `continent` on x-axis and `lifeExp` on y-axis

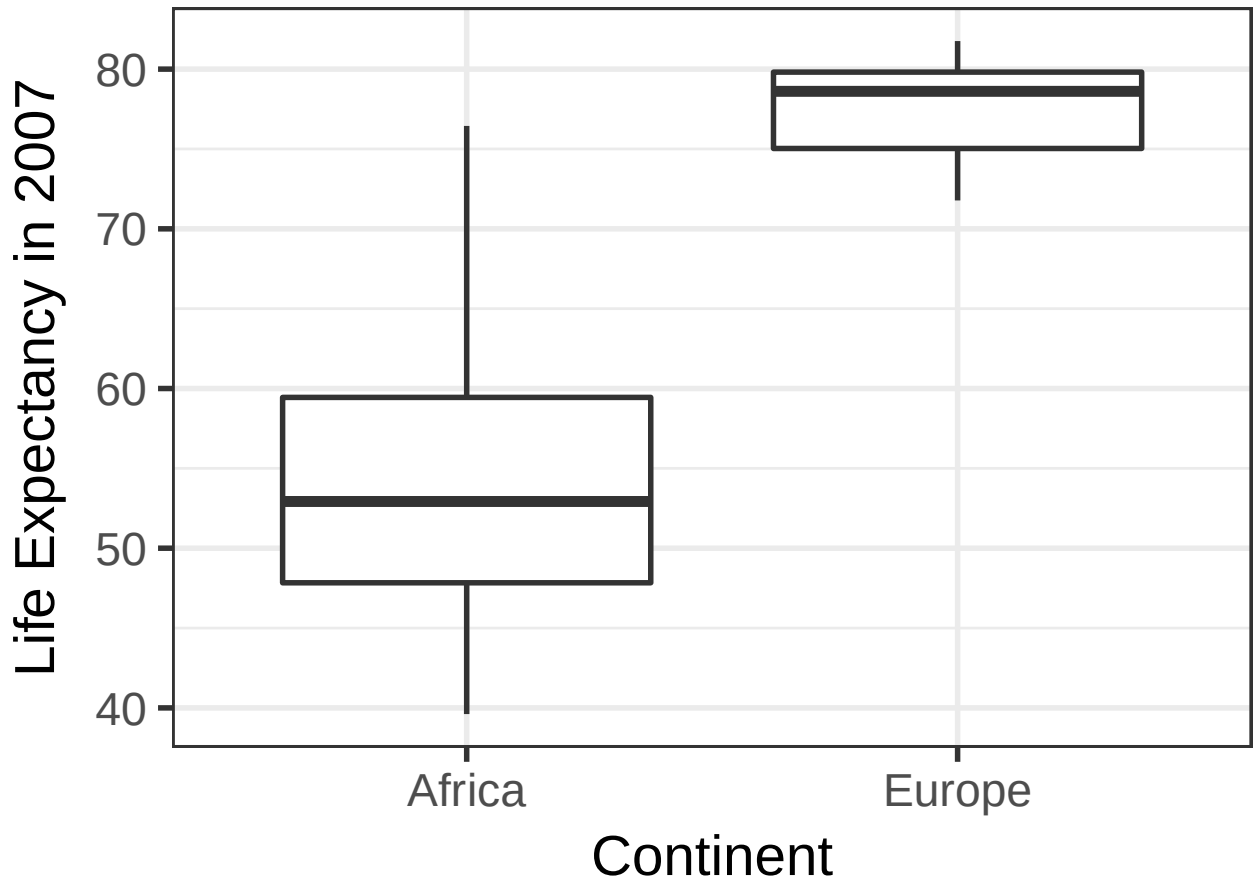
Solution

```
dta_europe_africa_2007 <- gapminder %>%
  filter(continent %in% c("Europe", "Africa") & year == 2007)

my_boxplot <- ggplot(dta_europe_africa_2007,
       aes(x = continent, y = lifeExp)) +
  geom_boxplot()
```

```
my_boxplot
```



```
my_boxplot +
  labs(x = "Continent", y = "Life Expectancy in 2007") +
  theme_bw()
```



General Advice

- Save plots as PDF (vector graphic and small):
- Set the **ggplot2** theme globally

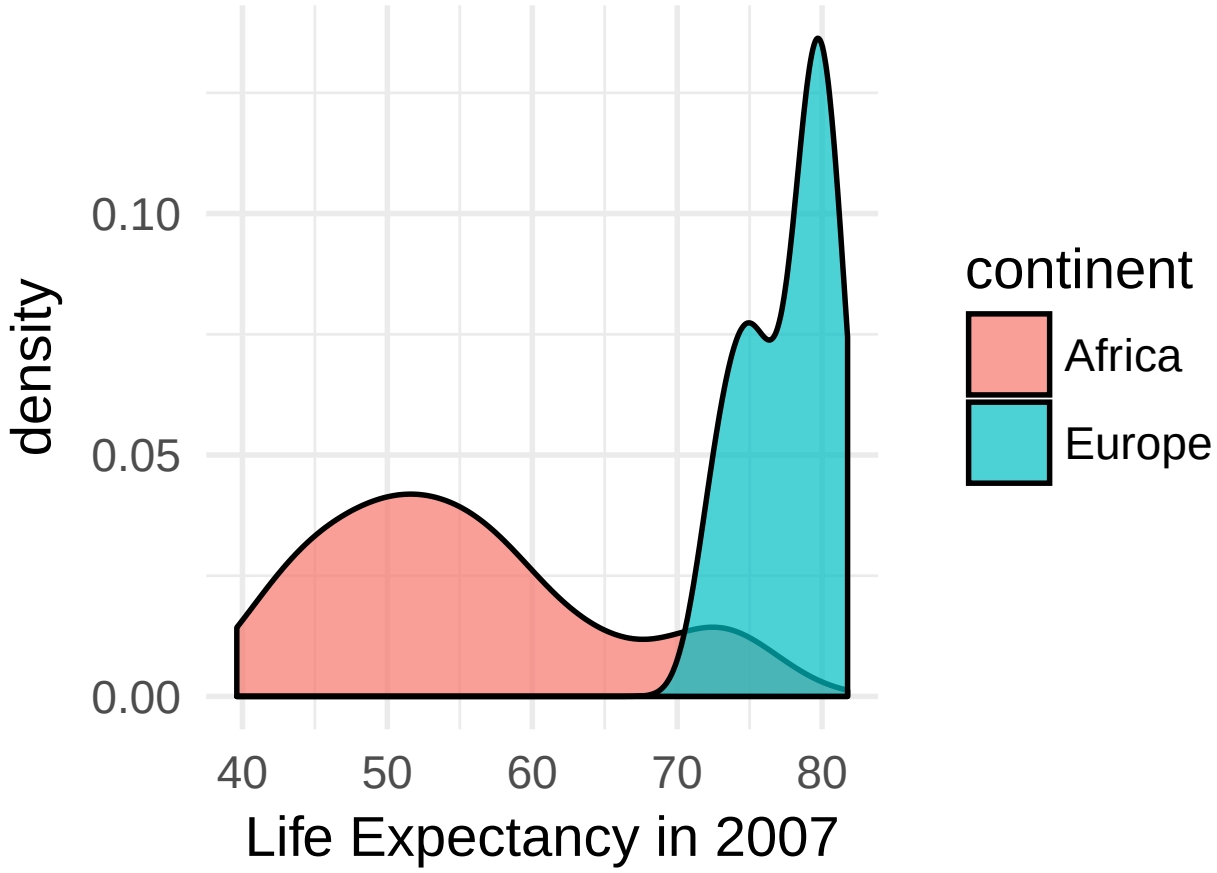
```
# example for saving a plot
ggsave("boxplot.pdf", my_boxplot, width = 5, height = 7)
```

```
# example for setting a theme
ggplot2::theme_set(theme_minimal())
```

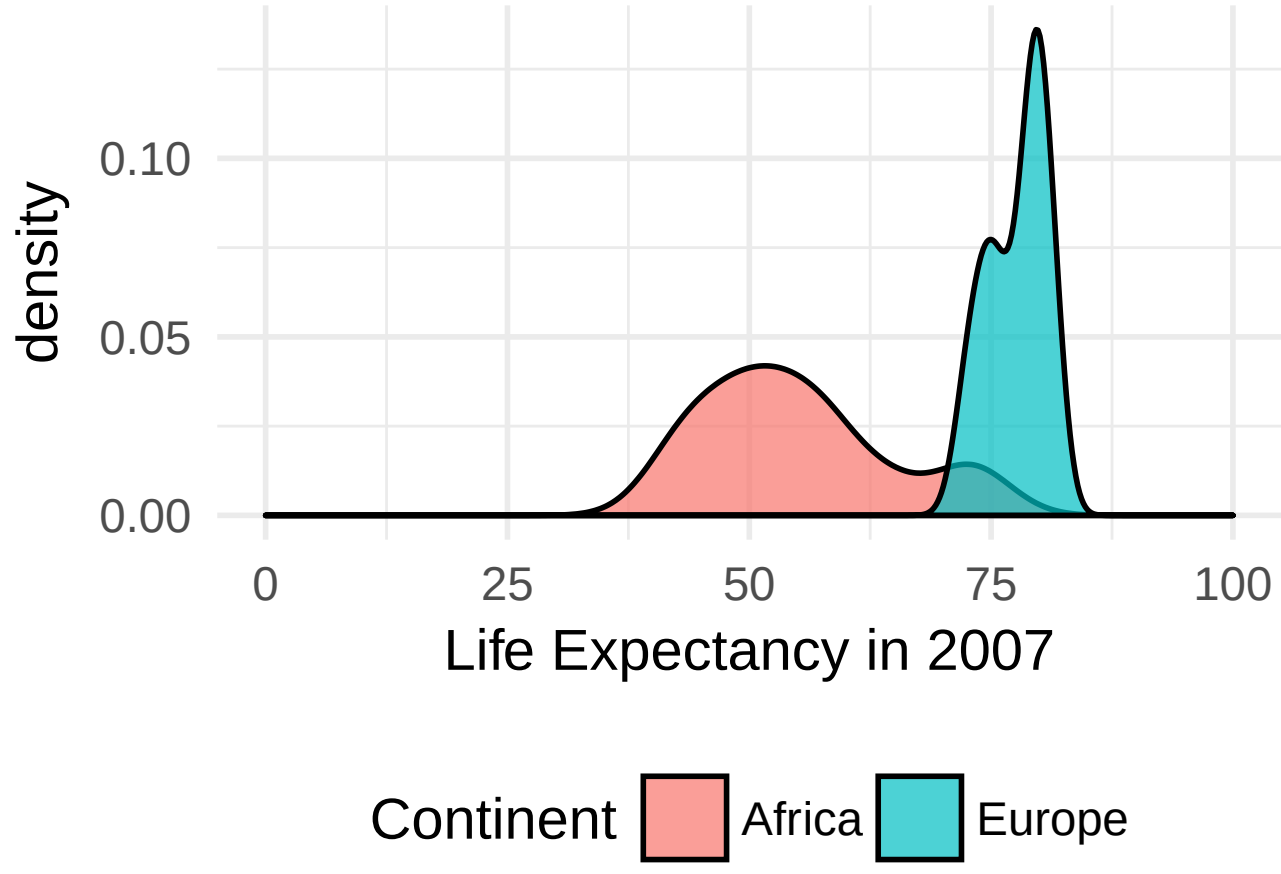
Density plots

```
plot_density <- ggplot(dta_europe_africa_2007,
       aes(x = lifeExp, fill = continent)) +
  geom_density(alpha = 0.7) +
  labs(x = "Life Expectancy in 2007")
```

```
plot_density
```



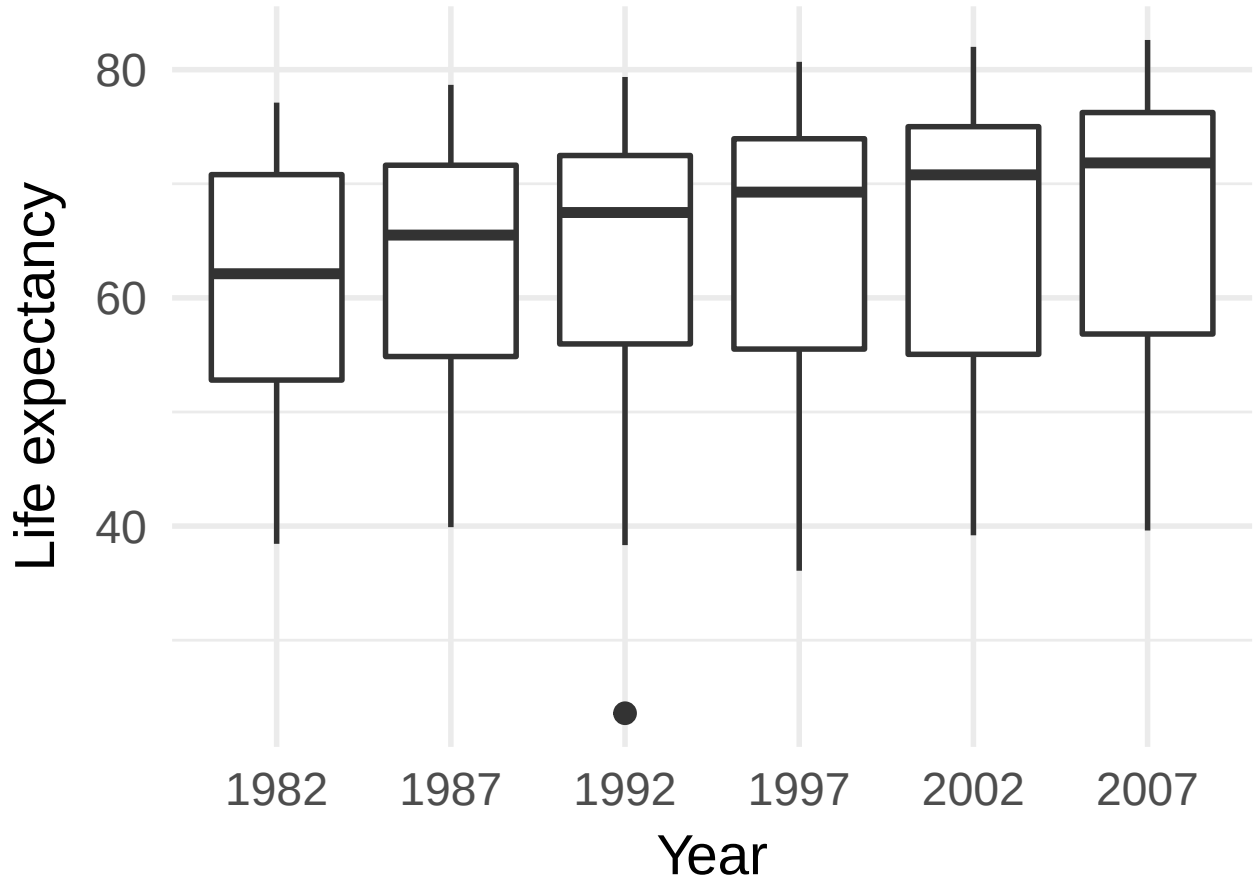
```r
plot_density +
  scale_x_continuous(limits = c(0, 100)) +
  scale_fill_discrete(name = "Continent") +
  theme(legend.position = "bottom")
```



Boxplots with groups

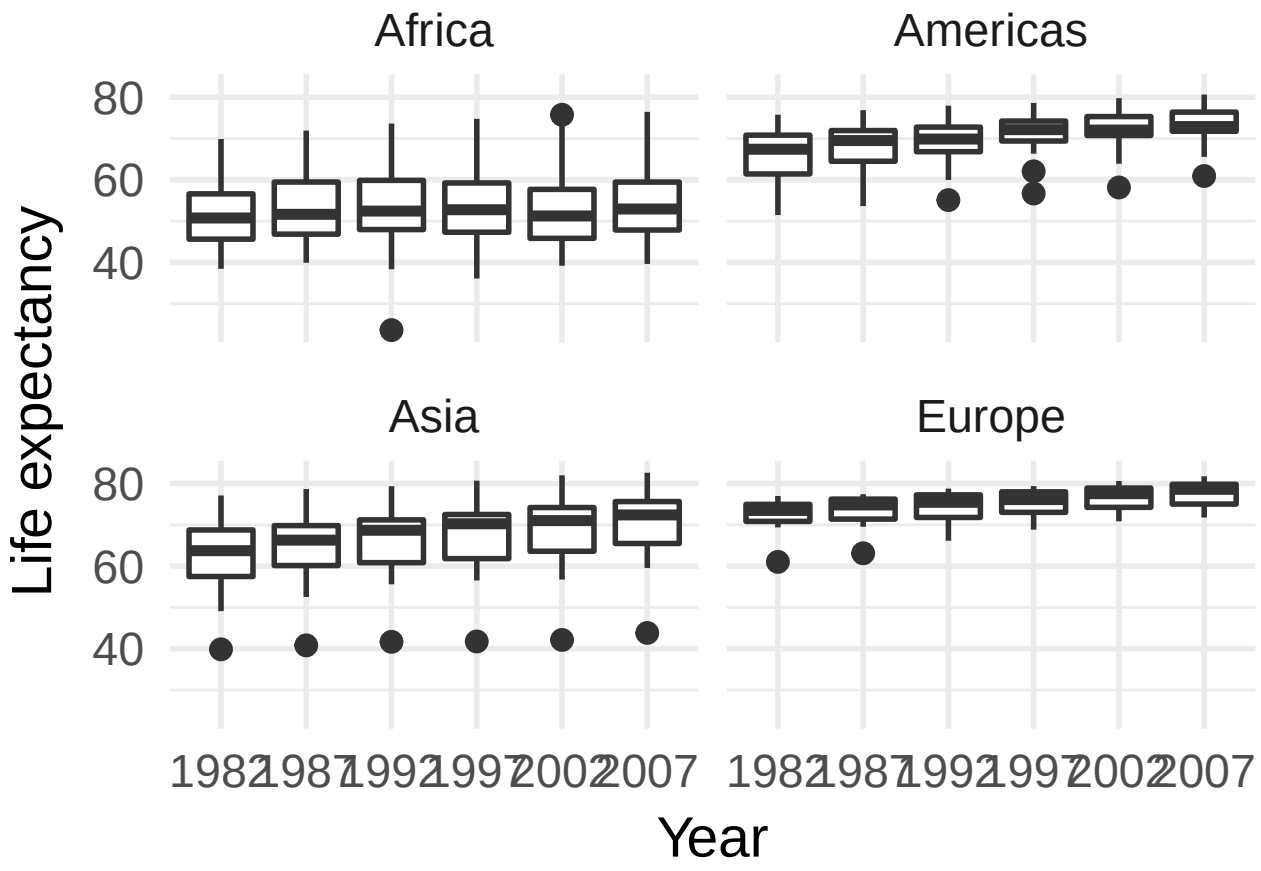
```
gapminder_post_1980 <- gapminder %>%
  filter(year > 1980 & continent != "Oceania")

plot_boxes <- ggplot(data = gapminder_post_1980,
       aes(x = as.factor(year), y = lifeExp)) +
  geom_boxplot() +
  labs(x = "Year", y = "Life expectancy")
```

```
plot_boxes
```



Add facets for a factor variable

```
plot_boxes + facet_wrap(~continent)
```



`arrange()` and `select()`

Arrange dataset

```
gap_arranged <- gapminder %>%
  arrange(lifeExp) # ascending

head(gap_arranged)
```

country	continent	year	lifeExp	pop	gdpPercap
Rwanda	Africa	1992	23.599	7290203	737.0686
Afghanistan	Asia	1952	28.801	8425333	779.4453
Gambia	Africa	1952	30.000	284320	485.2307
Angola	Africa	1952	30.015	4232095	3520.6103
Sierra Leone	Africa	1952	30.331	2143249	879.7877
Afghanistan	Asia	1957	30.332	9240934	820.8530

Arrange dataset

```
gap_arranged_desc <- gapminder %>%
  arrange(-lifeExp) # descending

head(gap_arranged_desc)
```

country	continent	year	lifeExp	pop	gdpPercap
Japan	Asia	2007	82.603	127467972	31656.07
Hong Kong, China	Asia	2007	82.208	6980412	39724.98
Japan	Asia	2002	82.000	127065841	28604.59
Iceland	Europe	2007	81.757	301931	36180.79
Switzerland	Europe	2007	81.701	7554661	37506.42
Hong Kong, China	Asia	2002	81.495	6762476	30209.02

Select variables

```
gapminder_select <- gapminder %>%
  select(year, country, lifeExp)

head(gapminder_select, 2)
```

year	country	lifeExp
1952	Afghanistan	28.801
1957	Afghanistan	30.332

Reorder variables within dataframe

```
gapminder_reordered <- gapminder %>%
  select(pop, year, everything())

head(gapminder_reordered, 2)
```

pop	year	country	continent	lifeExp	gdpPercap
8425333	1952	Afghanistan	Asia	28.801	779.4453
9240934	1957	Afghanistan	Asia	30.332	820.8530

Exercise

1. Select `country`, `continent`, `year`, `lifeExp` and `pop`
2. Arrange `gapminder` ascending by year *and* within year descending by population.

Solution

```
gapminder_arranged <- gapminder %>%
  select(-gdpPercap) %>%
  arrange(year, -pop)

# check View(gapminder_arranged)
```

`group_by`, `mutate()` and `summarise()`

```
df
```

colour	value
green	1
black	2
black	3
green	4
black	5
red	6

Group dataset and summarise within groups

```
df %>%
  summarise(sum = sum(value))
```

sum
21

```
df %>%
  group_by(colour) %>% # add grouping variable
  summarise(sum = sum(value))
```

colour	sum
black	10
green	5
red	6

```r
df %>%
  group_by(colour) %>%
  summarise(total = n(),
            sum = sum(value))
```

colour	total	sum
black	3	10
green	2	5
red	1	6

Exercise

1. Use `gapminder`.
2. Group by `year`.
3. Calculate the average life expectancy and the maximum GDP per year.

Solution

```
data_gap_summarised <- gapminder %>%
  group_by(year) %>%
  summarise(life_exp_mean = mean(lifeExp),
            gdp_max = max(gdpPercap))
```

```
data_gap_summarised
```

year	life_exp_mean	gdp_max
1952	49.05762	108382.35
1957	51.50740	113523.13
1962	53.60925	95458.11
1967	55.67829	80894.88
1972	57.64739	109347.87
1977	59.57016	59265.48
1982	61.53320	33693.18
1987	63.21261	31540.97
1992	64.16034	34932.92
1997	65.01468	41283.16
2002	65.69492	44683.98
2007	67.00742	49357.19

mutate creates new variables

```
names(gapminder)
```

```
## [1] "country"   "continent" "year"      "lifeExp"   "pop"
```

```
gapminder <- gapminder %>%
  group_by(year) %>%
  mutate(life_exp_mean = mean(lifeExp),
         diff_country_mean = lifeExp - life_exp_mean)
summary(gapminder$diff_country_mean)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -40.561   -9.583    1.293    0.000   10.240   23.612
```

Plotting

Exercises

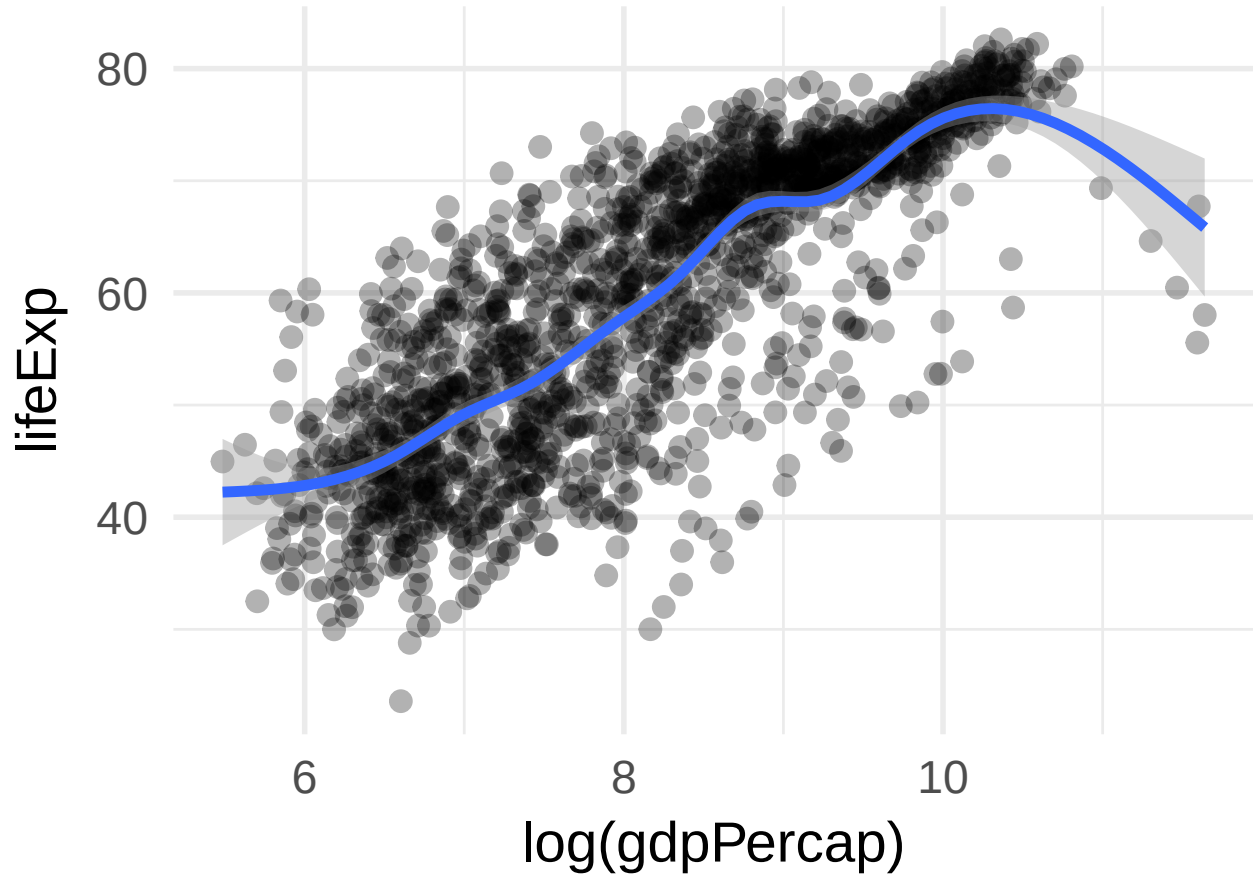
Look closely at the **ggplot2** cheat sheet and find appropriate plots to understand the relationships between:

1. `gdpPercap` and `lifeExp`;
2. `lifeExp` and `continent`
3. `year` and `lifeExp` by `continent`

Example: gdpPercap and lifeExp

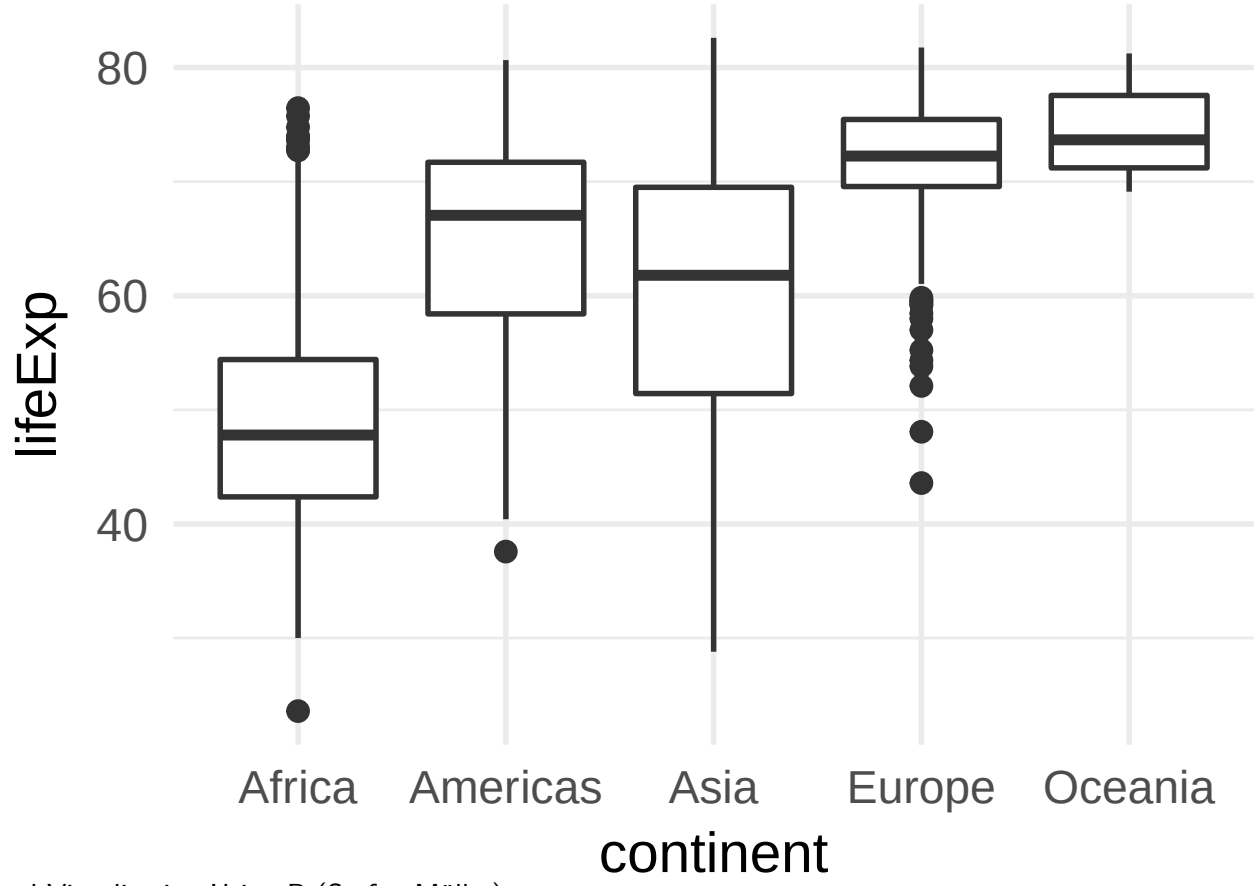
```
plot_gdp_life_exp <- ggplot(gapminder,
                            aes(x = log(gdpPercap),
                                y = lifeExp)) +
  geom_point(alpha = 0.3) +
  geom_smooth()
```

```
plot_gdp_life_exp
```



Example: `lifeExp` and `continent`

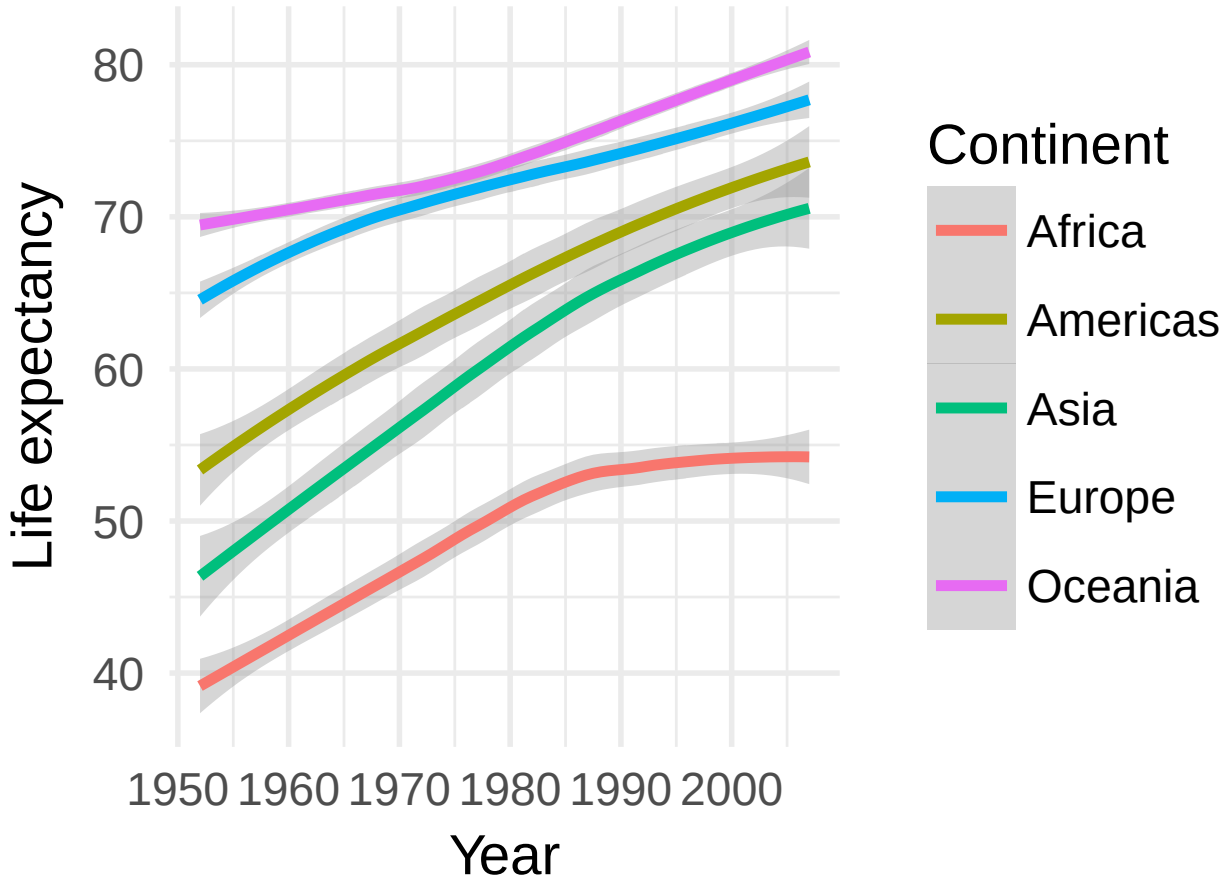
```
ggplot(data = gapminder, aes(x = continent, y = lifeExp)) +
  geom_boxplot()
```



Example: year and lifeExp by continent

```
plot_year_lifeExp <- ggplot(data = gapminder,
       aes(x = year, y = lifeExp,
           colour = continent)) +
  geom_smooth() +
  scale_colour_discrete(name = "Continent") +
  labs(x = "Year", y = "Life expectancy")
```

```
plot_year_lifeExp
```



Merge data with `..._join`

Load ParlGov datasets

```
url_cabinets <- "http://www.parlgov.org/static/data/development-
parlgov_cabinets <- read.csv(url(url_cabinets))

dim(parlgov_cabinets)
```

```
## [1] 10903    19
```

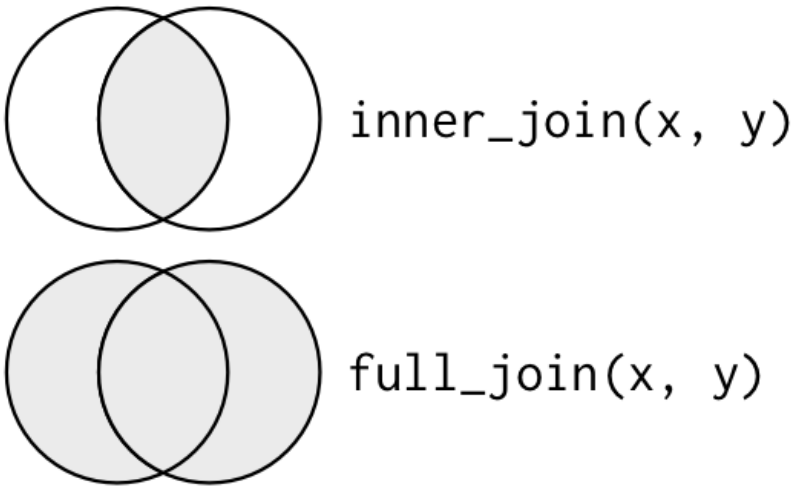
```
url_elections <- "http://www.parlgov.org/static/data/development
parlgov_elections <- read.csv(url(url_elections))

dim(parlgov_elections)
```

```
## [1] 7808   16
```

Select a subset of variables

```
parlgov_elections_small <- parlgov_elections %>%
  select(country_name, election_date, party_id, vote_share)
```

Join options



Join ParlGov data

```
parlgov_joined <- left_join(parlgov_cabinets,
                            parlgov_elections_small,
                            by = c("country_name",
                                   "election_date",
                                   "party_id"))
```

Conclusion

- `select`, `filter`, `arrange`, `mutate`, `group_by` and `summarise` should solve many problems.
- Import data with `readr` or `rio`.
- Useful resources:
  - Cheat sheets: https://www.rstudio.com/resources/cheatsheets/
  - Wickham and Grolemund (2016): R For Data Science. http://r4ds.had.co.nz
  - Answers on Stack Overflow: http://stackoverflow.com