Visualizing health data in R and RStudio

Corinne Riddell

August 29, 2022

Learning objectives for today

1. To put to use the `dplyr` commands from last lecture
2. To make beautiful plots using the `ggplot2` package

Readings

- Online resource: Creating a `ggplot`(See sections 3.2.2-3.2.3)

Life expectancy in the United States by race and gender, 1969-2013

These data are partial results from a study that I did on the difference in life expectancy between Black and White men and women in the United States over time.

A subset of the results have been stored in the datahub folder for today's lecture as a comma separated value (CSV) file.

Do you remember which function to use to import CSV data into R?

`readr`'s `read_csv()` to import these data

```r
library(readr)
le_data <- read_csv("./data/Life-expectancy-by-state-long.csv")
```

```
## Rows: 7200 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): state, stabbrs, sex, Census_Region, Census_Division, race
## dbl (2): year, LE
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Four functions to get to know your dataset

Function 1

```r
head(le_data)
```

```
## # A tibble: 6 x 8
##   state   stabbrs  year sex    Census_Region Census_Division       LE race
##   <chr>   <chr>   <dbl> <chr>  <chr>         <chr>              <dbl> <chr>
## 1 Alabama AL       1969 Female South         East South Central  75.8 white
## 2 Alabama AL       1969 Male   South         East South Central  66.6 white
## 3 Alabama AL       1970 Female South         East South Central  75.9 white
## 4 Alabama AL       1970 Male   South         East South Central  66.7 white
## 5 Alabama AL       1971 Female South         East South Central  76.2 white
## 6 Alabama AL       1971 Male   South         East South Central  66.9 white
```

Four functions to get to know your dataset

Function 2

```r
dim(le_data)
```

```
## [1] 7200    8
```

Four functions to get to know your dataset

Function 3

```r
names(le_data)
```

```
## [1] "state"           "stabbrs"         "year"            "sex"
## [5] "Census_Region"   "Census_Division" "LE"              "race"
```

Four functions to get to know your dataset

Function 4

```r
str(le_data)
```

```
## spec_tbl_df [7,200 x 8] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ state          : chr [1:7200] "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ stabbrs        : chr [1:7200] "AL" "AL" "AL" "AL" ...
##  $ year           : num [1:7200] 1969 1969 1970 1970 1971 ...
##  $ sex            : chr [1:7200] "Female" "Male" "Female" "Male" ...
##  $ Census_Region  : chr [1:7200] "South" "South" "South" "South" ...
##  $ Census_Division: chr [1:7200] "East South Central" "East South Central" "East South Central" "Eas
##  $ LE             : num [1:7200] 75.8 66.6 75.9 66.7 76.2 ...
##  $ race           : chr [1:7200] "white" "white" "white" "white" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   state = col_character(),
##   ..   stabbrs = col_character(),
##   ..   year = col_double(),
##   ..   sex = col_character(),
##   ..   Census_Region = col_character(),
##   ..   Census_Division = col_character(),
##   ..   LE = col_double(),
##   ..   race = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

A new useful function: View()

```
View(le_data)
```

`View()` opens the data viewer pane in RStudio. It does not print anything in the data console.

Life expectancy for White men in California

Make a scatter plot of the life expectancy for White men in California over time.

Since the dataset contains 39 states across two genders and two races, first use a function to subset the data to contain only White men in California.

Which function from last lecture do we need?

- `mutate()`, `select()`, `filter()`, `rename()`, or `arrange()`?

`dplyr`'s `filter()` to select a subset of rows

```
library(dplyr)
```

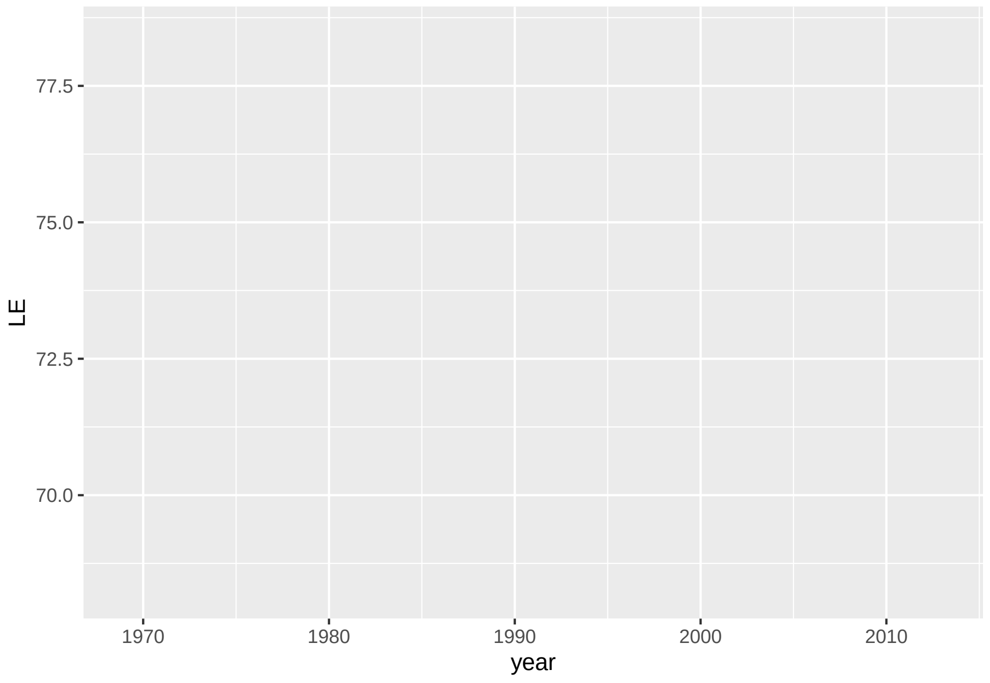
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
wm_cali <- le_data %>% filter(state == "California",
                              sex == "Male",
                              race == "white")

#this is equivalent:
wm_cali <- le_data %>% filter(state == "California" & sex == "Male" & race == "white")
```

First step to building a `ggplot()`: set up the canvas

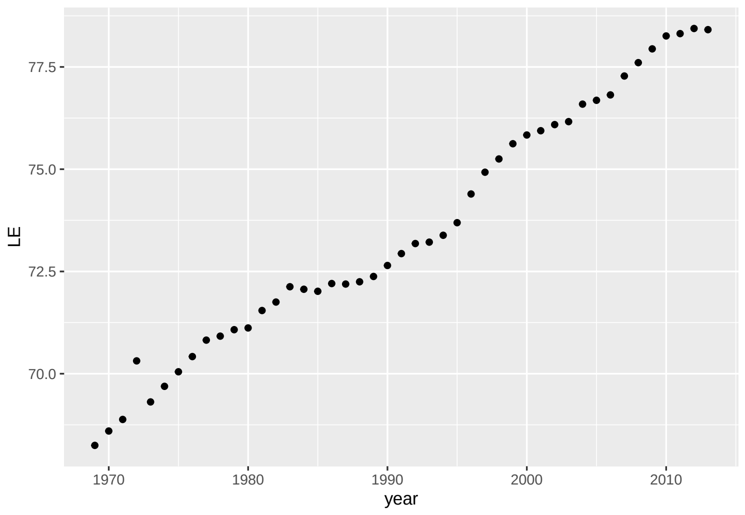
- The second line of code below specifies the `data` set and what goes on the `x` and `y` axes

```
library(ggplot2)
ggplot(data = wm_cali, aes(x = year, y = LE))
```



Second step to building a `ggplot()`: tell `ggplot` how to plot the data

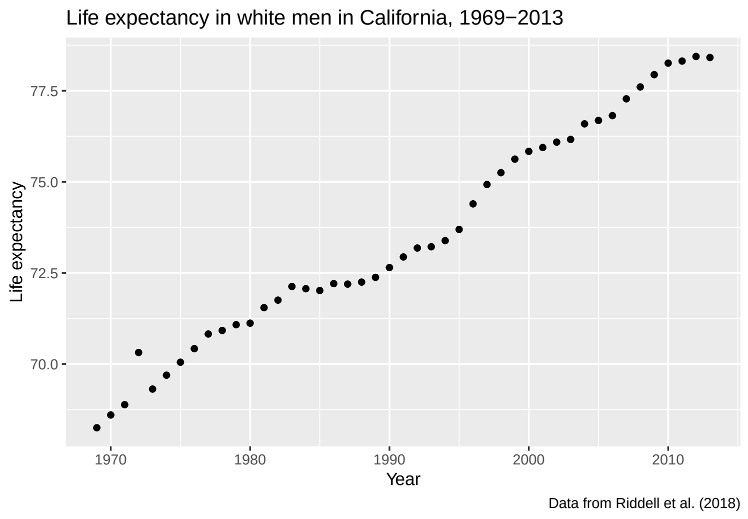
```
ggplot(data = wm_cali, aes(x = year, y = LE)) + geom_point()
```



- `geom_point()` tells ggplot to use points to plot these data

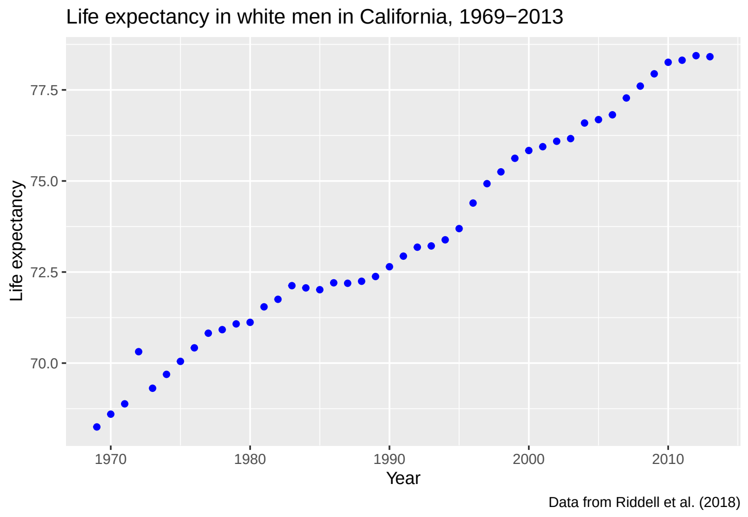
`labs()` to add a title, a caption, and modify x and y axes titles

```r
ggplot(data = wm_cali, aes(x = year, y = LE)) + geom_point() +
  labs(title = "Life expectancy in white men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



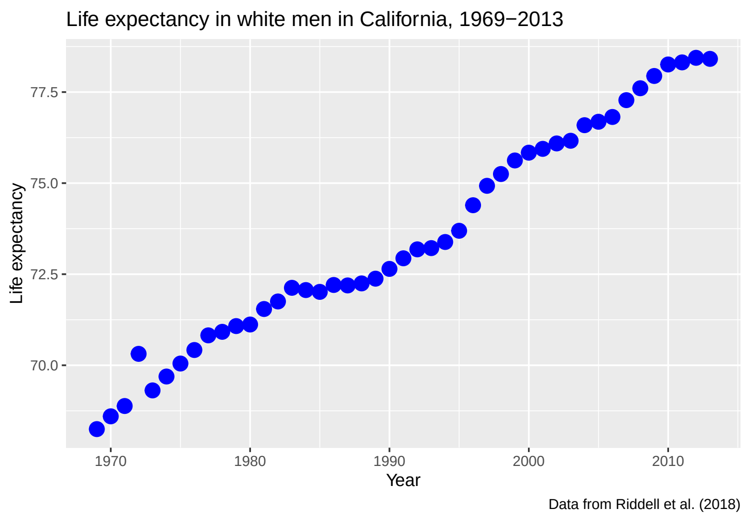
`col` controls the color of geom_point()

```r
ggplot(data = wm_cali, aes(x = year, y = LE)) + geom_point(col = "blue") +
  labs(title = "Life expectancy in white men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



size controls the size of geom_point()

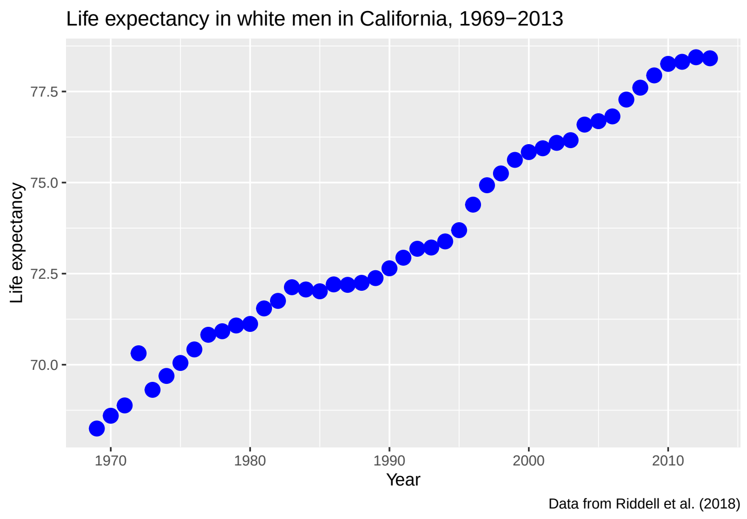
```r
ggplot(data = wm_cali, aes(x = year, y = LE)) + geom_point(col = "blue", size = 4) +
  labs(title = "Life expectancy in white men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



Line plot rather than scatter plot

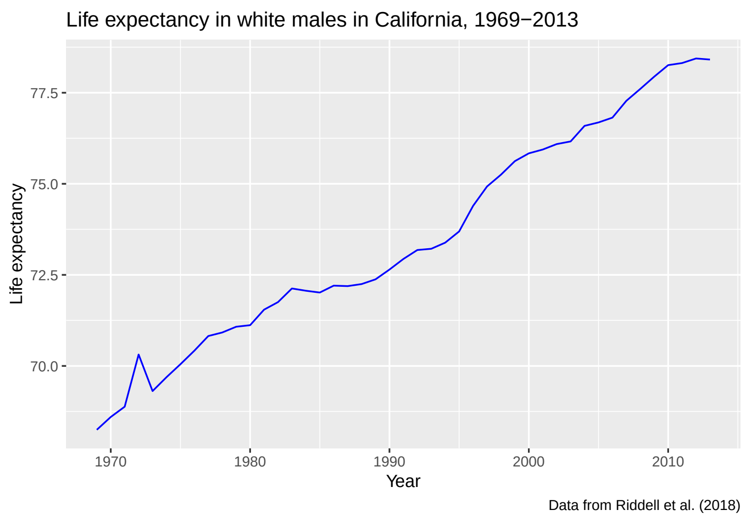
What if we wanted to make these data into a line plot instead. What part of the code should change?

```r
ggplot(data = wm_cali, aes(x = year, y = LE)) +
  geom_point(col = "blue", size = 4) +
  labs(title = "Life expectancy in white men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



`geom_line()` to make a line plot

```r
ggplot(data = wm_cali, aes(x = year, y = LE)) + geom_line(col = "blue") +
  labs(title = "Life expectancy in white males in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



Life expectancy for White and Black men in California

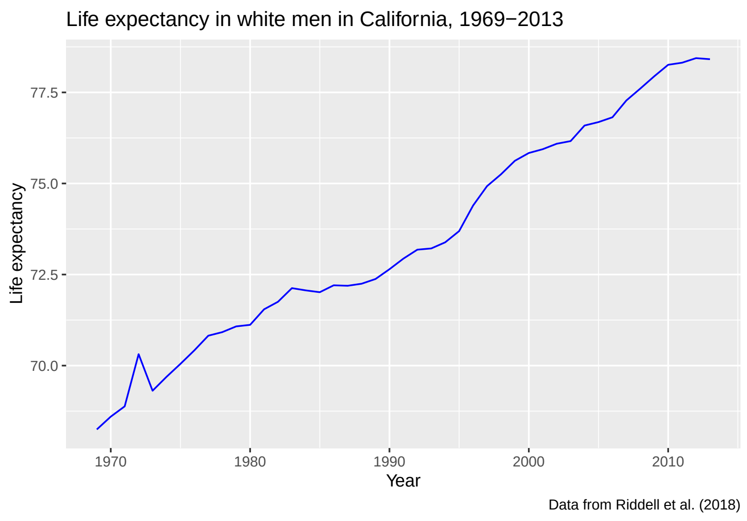
What do we need to change to make a separate line for both Black and White men?

First, update the `filter()`

```r
wbm_cali <- le_data %>% filter(state == "California",
                               sex == "Male")
```

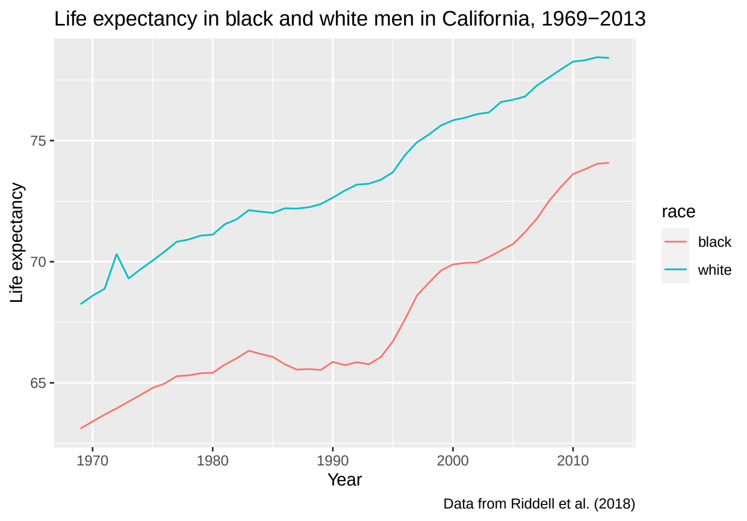
Look at the previous code and output first

```r
ggplot(data = wm_cali, aes(x = year, y = LE)) + geom_line(col = "blue") +
  labs(title = "Life expectancy in white men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



And change it to link color to race

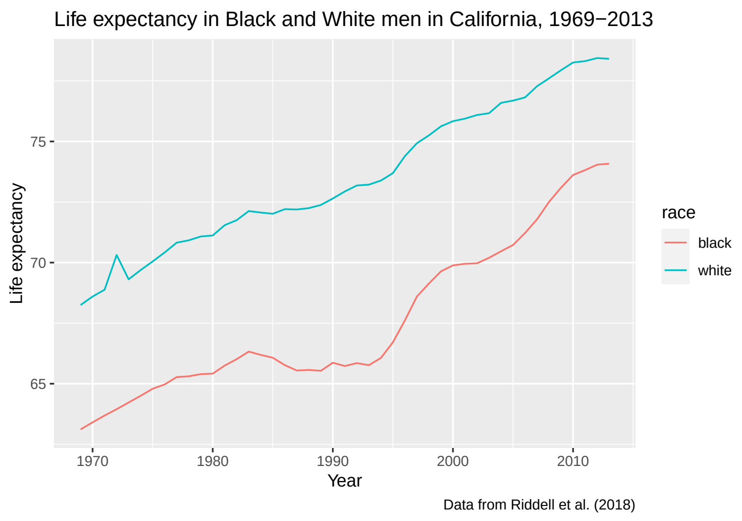
```
ggplot(data = wbm_cali, aes(x = year, y = LE)) + geom_line(aes(col = race)) +
  labs(title = "Life expectancy in black and white men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



Always use the aes() function to link a plot feature to a variable in your data frame

The operative word is *link*. Whenever you want to link something about how the plot looks to a variable in the data frame, you need to *link* these items inside the `aes()` function:

```
ggplot(data = wbm_cali, aes(x = year, y = LE)) + geom_line(aes(col = race)) +
  labs(title = "Life expectancy in Black and White men in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



The `aes()` function

- What else was added to the plot when you used the `aes()` function?

The `aes()` function

- What else was added to the plot when you used the `aes()` function?
  - A legend was added showing the link between the line color and the data frame's race variable

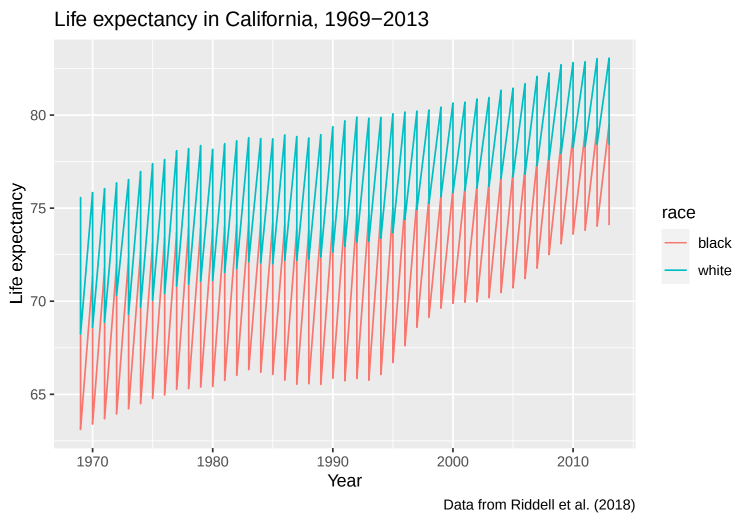
What if we also wanted to look at women?

What if we also wanted to look at women?

```
cali_data <- le_data %>% filter(state == "California")
```

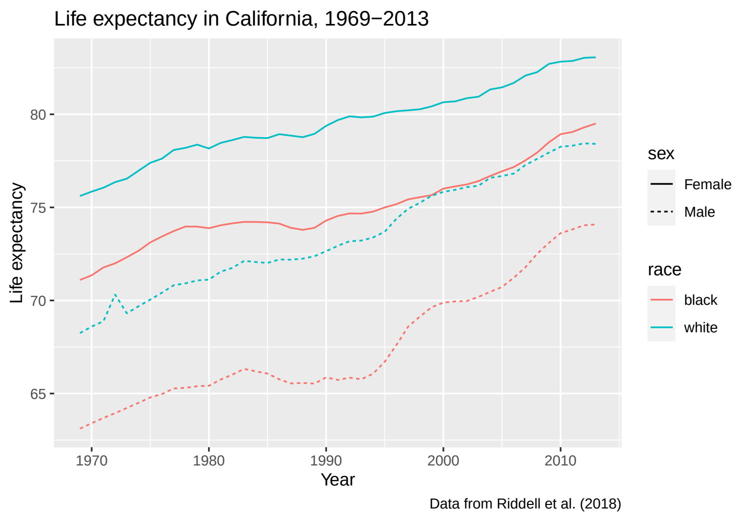
What is wrong with this plot?

```
ggplot(data = cali_data, aes(x = year, y = LE)) + geom_line(aes(col = race)) +
  labs(title = "Life expectancy in California, 1969-2013",
        y = "Life expectancy",
        x = "Year",
        caption = "Data from Riddell et al. (2018)")
```



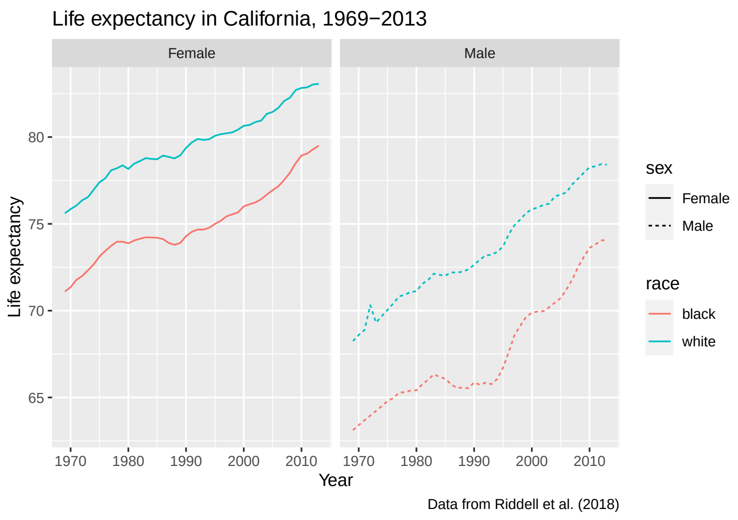
Use `lty()` to link line type to sex

```
ggplot(data = cali_data, aes(x = year, y = LE)) + geom_line(aes(col = race, lty = sex)) +
  labs(title = "Life expectancy in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)")
```



Use `facet_wrap()` to make separate plots for a specified variable

```
ggplot(data = cali_data, aes(x = year, y = LE)) +
  geom_line(aes(col = race, lty = sex)) +
  labs(title = "Life expectancy in California, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)") +
  facet_wrap(~ sex)
```



Compare two states

How do we update the `filter` to include data from California and New York?

Compare two states

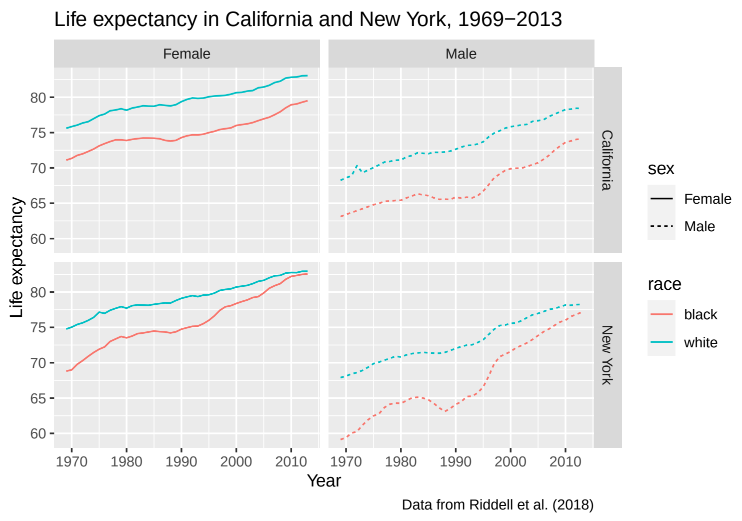
```
updated_data <- le_data %>% filter(state %in% c("California", "New York"))
```

Let's write the code together

```
#to fill in during class
```

Let's write the code together

```
ggplot(data = updated_data, aes(x = year, y = LE)) +
  geom_line(aes(col = race, lty = sex)) +
  labs(title = "Life expectancy in California and New York, 1969-2013",
       y = "Life expectancy",
       x = "Year",
       caption = "Data from Riddell et al. (2018)") +
  facet_grid(state ~ sex)
```



Check your understanding!

So far

- `geom_point()` to make scatter plots
- `geom_line()` to make line plots
- `col = "blue"`, `size = 2`, `lty = 2`, to change color, size and line type of the `geom`
- `aes(col = race)` to *link* color to race
- `aes(lty = sex)` to *link* line type to sex
- `facet_wrap(~ var1)` to make separate plots for different levels of one variable
- `facet_grid(var1 ~ var2)` to make separate plots for combinations of levels of two variables

What if we wanted to make a histogram...

...of life expectancy of white males in 2013?

Before you code, try and visualize what the histogram will show

- What is on the x axis? What is on the y axis?

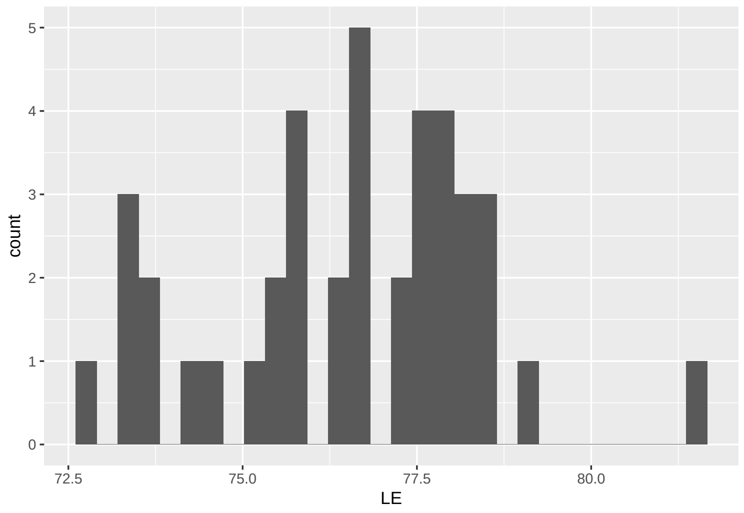
Update the `filter`

```
wm_data <- le_data %>% filter(year == 2013, sex == "Male", race == "white")
```

`geom_histogram()` to make histograms

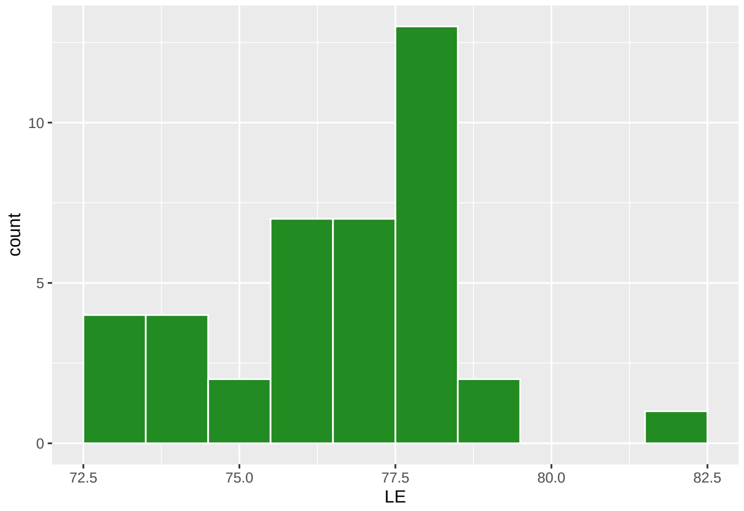
```
ggplot(dat = wm_data, aes(x = LE)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



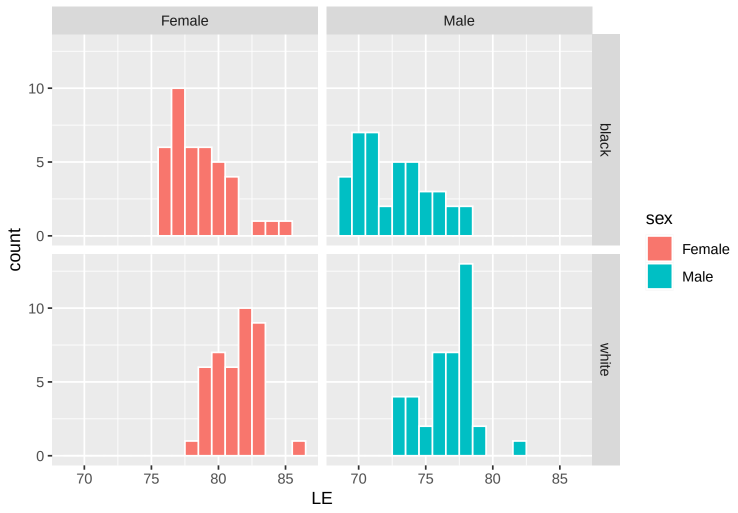
Use `fill` to change the *fill* of the histogram and `binwidth` to specify the bin's width

```r
wm_data <- le_data %>% filter(year == 2013, sex == "Male", race == "white")

ggplot(dat = wm_data, aes(x = LE)) +
  geom_histogram(binwidth = 1, col = "white", fill = "forest green")
```



Apply some of our new skills

```r
data_2013 <- le_data %>% filter(year == 2013)

ggplot(dat = data_2013, aes(x = LE)) +
  geom_histogram(binwidth = 1, col = "white", aes(fill = sex)) +
  facet_grid(race ~ sex)
```



Recap: What functions did we learn?

1. `ggplot()`
    - `geom_scatter()`
    - `geom_line()`
    - `geom_histogram()`
    - `aes()` to link aesthetics to variables in our data frame
    - `facet_wrap(~ var1)`, `facet_grid(var1 ~ var2)`
    - `labs(title = "Main", y = "y axis", x = "x axis", caption = "below plot")`

Recap: What arguments were useful?

2. `ggplot()`
    - `col`
    - `size`
    - `lty`

A common `dplyr` mistake to avoid!

With dplyr, the pipe operator goes at the end of the line, not the beginning:

Right way:

```
data_2013 <- le_data %>%
  filter(year == 2013)

#or

data_2013 <- le_data %>% filter(year == 2013)
```

Wrong way:

```
data_2013 <- le_data
%>% filter(year == 2013)
```

A common `ggplot` mistake to avoid!

With ggplot2, the "+"" operator goes at the end of the line, not the beginning:

Right way:

```
ggplot(dat = data_2013, aes(x = LE)) +
  geom_histogram(binwidth = 1, col = "white", aes(fill = sex)) +
  facet_grid(race ~ sex)
```

Wrong way:

```
ggplot(dat = data_2013, aes(x = LE))
  + geom_histogram(binwidth = 1, col = "white", aes(fill = sex))
  + facet_grid(race ~ sex)
```

How to get help with code

- Ask questions during discussion, GSI office hours, or on bCourses discussion forum. Use the appropriate thread!
- Develop your online search skills. For example if you have a `ggplot2` question, begin your google search with "r ggplot" and then describe your issues, e.g., "r ggplot how do I make separate lines by a second variable".
- The most common links that will appear are:
  - https://stackoverflow.com: Crowd-sourced answers that have been upvoted. The top answer is often the best one.
  - https://ggplot2.tidyverse.org/: The official ggplot2 webpage is very helpful.
  - https://community.rstudio.com/: The RStudio community page.
  - https://rpubs.com/: Web pages made by R users that often contain helpful tutorials.

We only skimmed the surface!

- Here is some extra material for those of you who love data visualization. This material won't be tested.
  - RStudio ggplot2 cheatsheet
  - Kieran Healy's data visualization book