Exploring gapminder

Practice notebook for DSC 205 Spring 2022

1 Time series plots

1.1 Concept

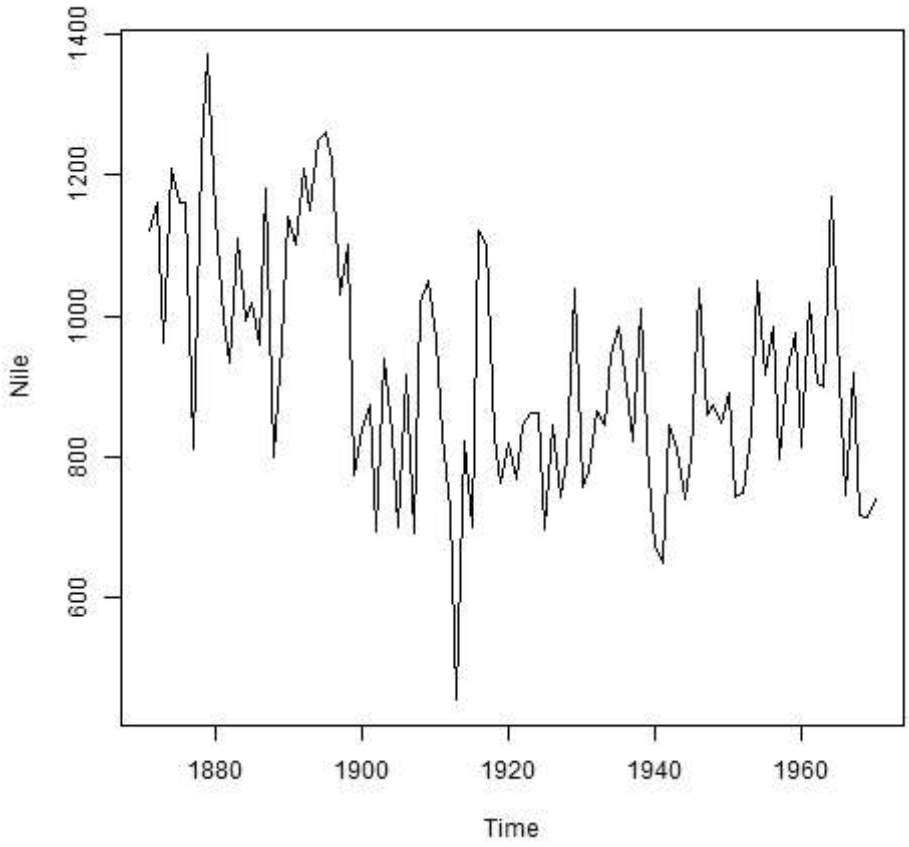
- Time series plots have time in the x-axis and an outcome or measurement of interest on the y-axis.
- Tableau, the data analysis platform, has interesting resources on time series plots, which are very popular in finance.
- Tableau is featured in the Data Visualization course (DSC 305).

1.2 Classic first - the Nile

- `[ ]`

  We begin with a classic - the univariate (= 1 variable) `Nile` data set, which is a `time series` object. Plot this using the Base-R `plot` function and as line type: `type="l"`.

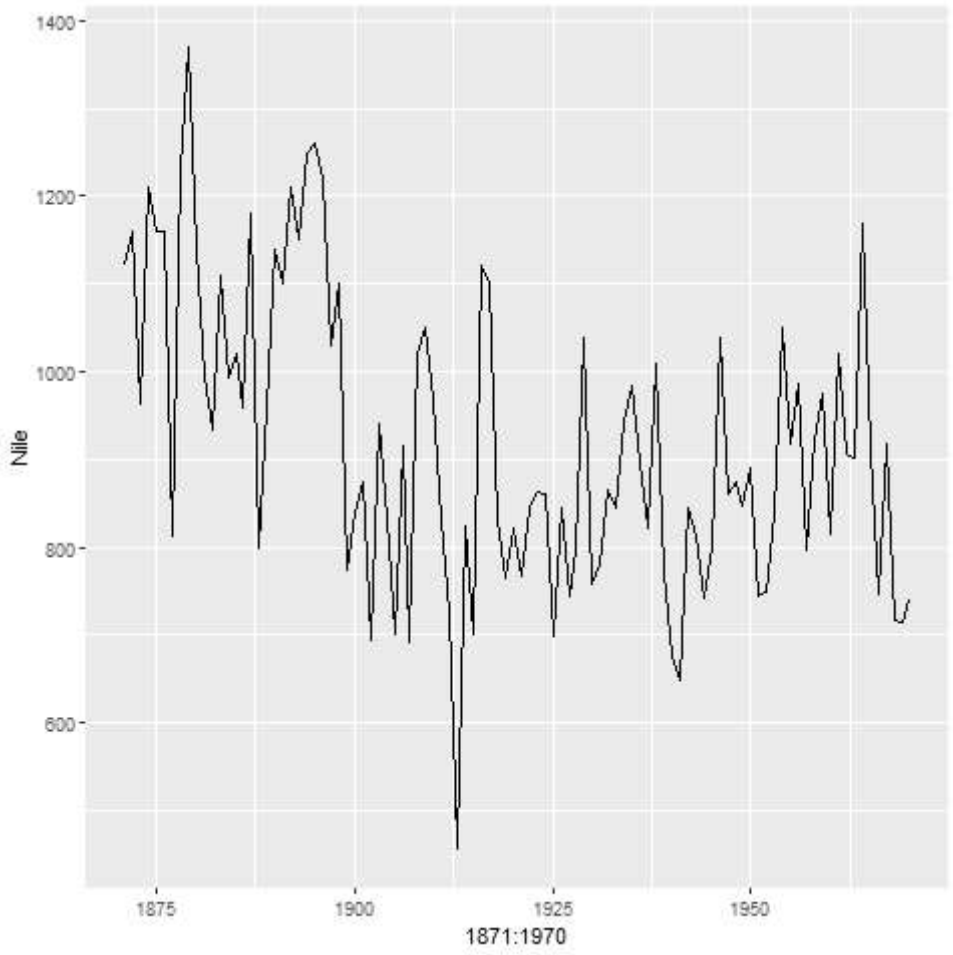
```
plot(Nile, type="l")
```



- [ ]

  Now, plot `Nile` using `ggplot`. For the scatterplot and line plot, you need to specify both `aes` arguments, and you need to feed `ggplot` with a `data.frame`.

```
data.frame(Nile) %>%
  ggplot(aes(x=1871:1970,y=Nile)) +
  geom_line()
```



- [ ]

  Are these last two plots identical? `plot` results cannot be stored as R objects, but `ggplot` results can. Compare the plots.

  1. store the `ggplot` in an object `g`
  2. run `identical` on `g` and the `plot` command from 1.

  /Tip: you cannot assign a pipe to a variable, so you'll have to find another way of feeding `gapminder` to the `ggplot` function.

```
g <- ggplot(data=data.frame(Nile), aes(x=1871:1970,y=Nile))+geom_line()
identical(g,plot(Nile,type="l"))
```

```
[1] FALSE
```

- [ ]

  `g` is a completely different R object from `plot(Nile)`. Check their `class`.

```
class(plot(Nile))
class(g)
```

```
[1] "NULL"
[1] "gg"     "ggplot"
```

1.3 US fertility rates over time: scatterplot

- [ ]

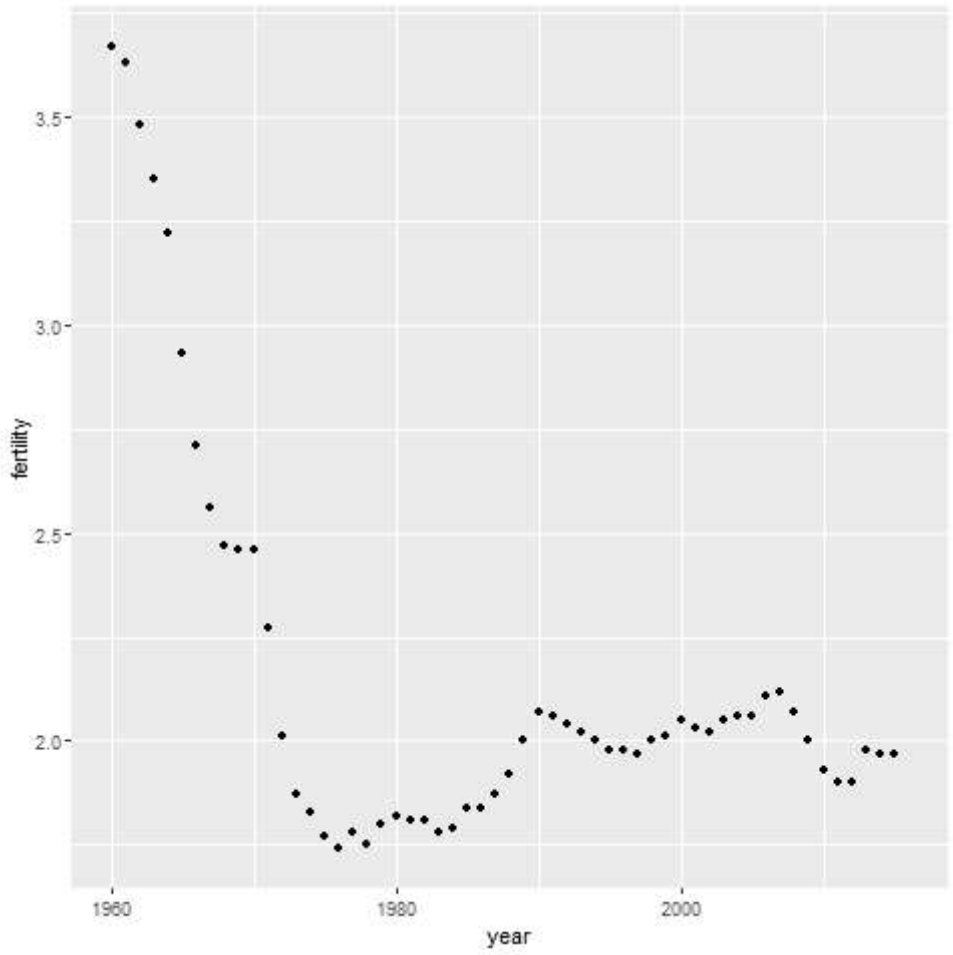
Example: US `fertility` rates from 1960 to 2012.

1. Filter the `country` `United States` out of `gapminder`.
2. Plot `year` vs. `fertility` as a scatterplot.

This pipeline command has three parts:

1. the dataset
2. the filter
3. the plot - `aes` data mapping and geometry.

```
gapminder %>%
  filter(country == "United States") %>%
  ggplot(aes( x = year, y = fertility)) +
  geom_point()
```

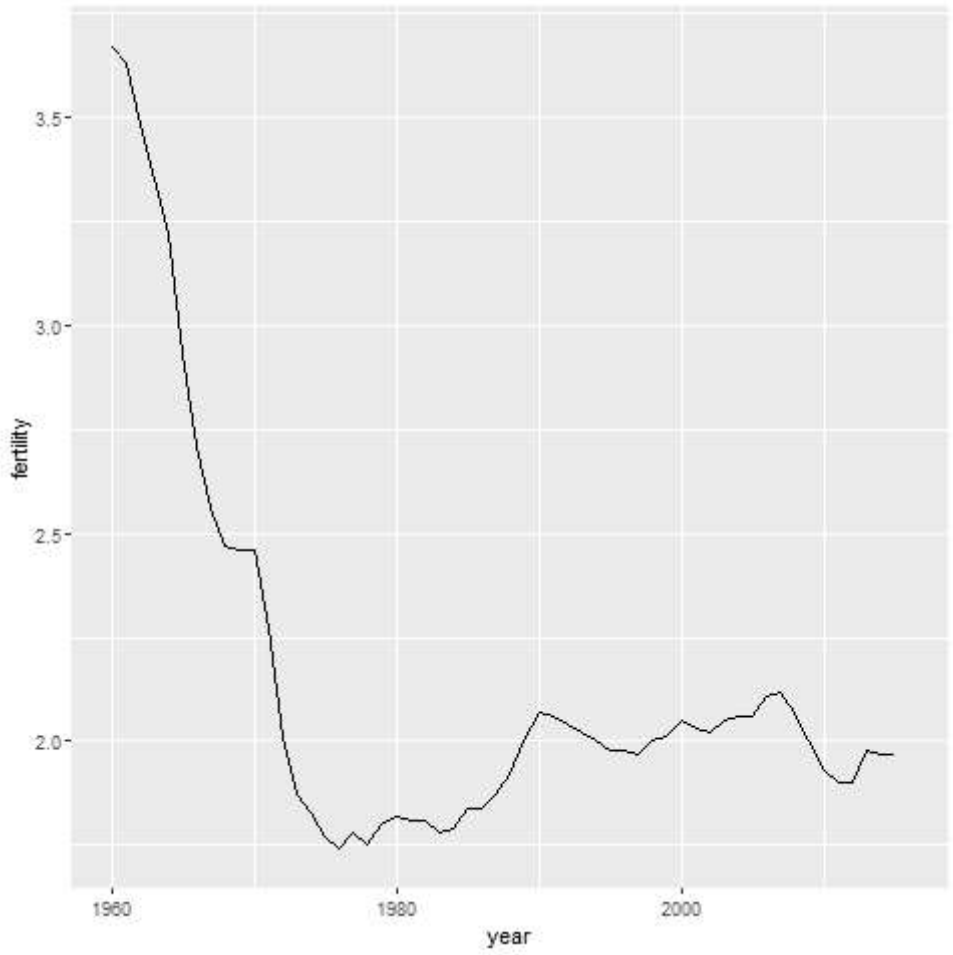


1.4 US fertility rates over time: lineplot

- [ ]

  Turn the plot into a **line plot**. Lines are easier to follow than scattered points.

```
gapminder %>%
  filter(country == "United States") %>%
  ggplot(aes( x = year, y = fertility)) +
  geom_line()
```



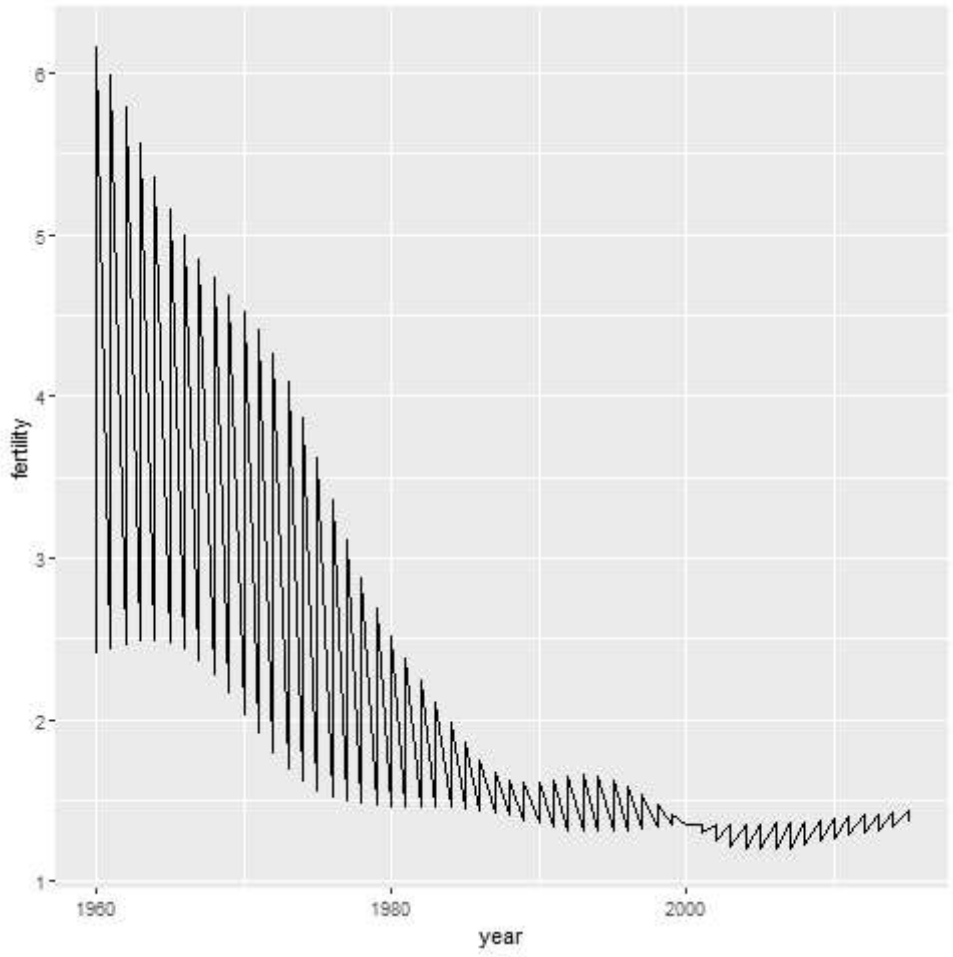
1.5 Fertility rates over time for two countries: grouping

- [ ]

  Let's look at **two countries** at once.

  1. Define a `countries` vector with `South Korea` and `Germany` in it.
  2. Filter both countries out of `gapminder`
  3. Add the condition `!is.na(fertility)` to the filter[1]
  4. Make a line plot of `year` vs. `fertility`

```
countries <- c("South Korea", "Germany")
gapminder %>%
  filter(country %in% countries & !is.na(fertility)) %>%
  ggplot(aes(x = year, y = fertility)) +
  geom_line()
```

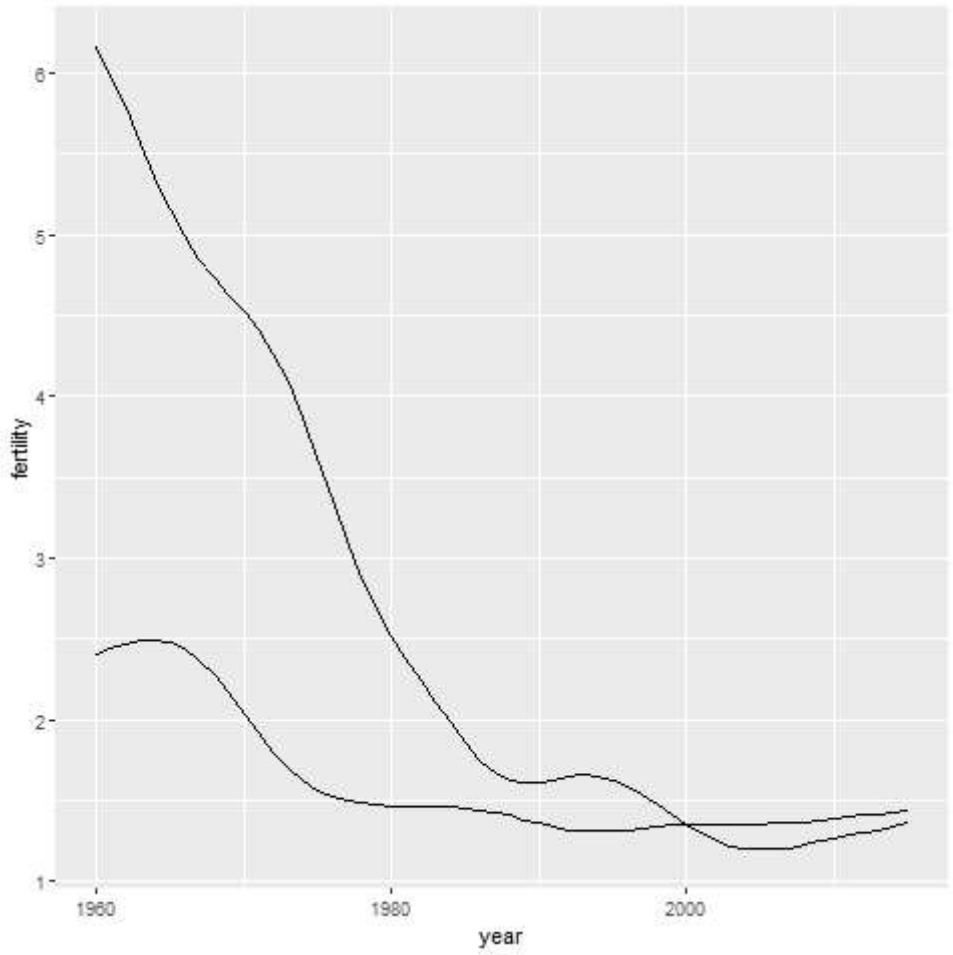


- Bummer! We haven't told `ggplot` anything about separating these data, so they're all connected by the same line.
- `[ ]`

  To **separate** the data from different countries, we use the `group` attribute in the `aes` data mapping function: repeat the last command, and add `group = country` as an argument to `aes()`.

  You can learn more about grouping aesthetics here.

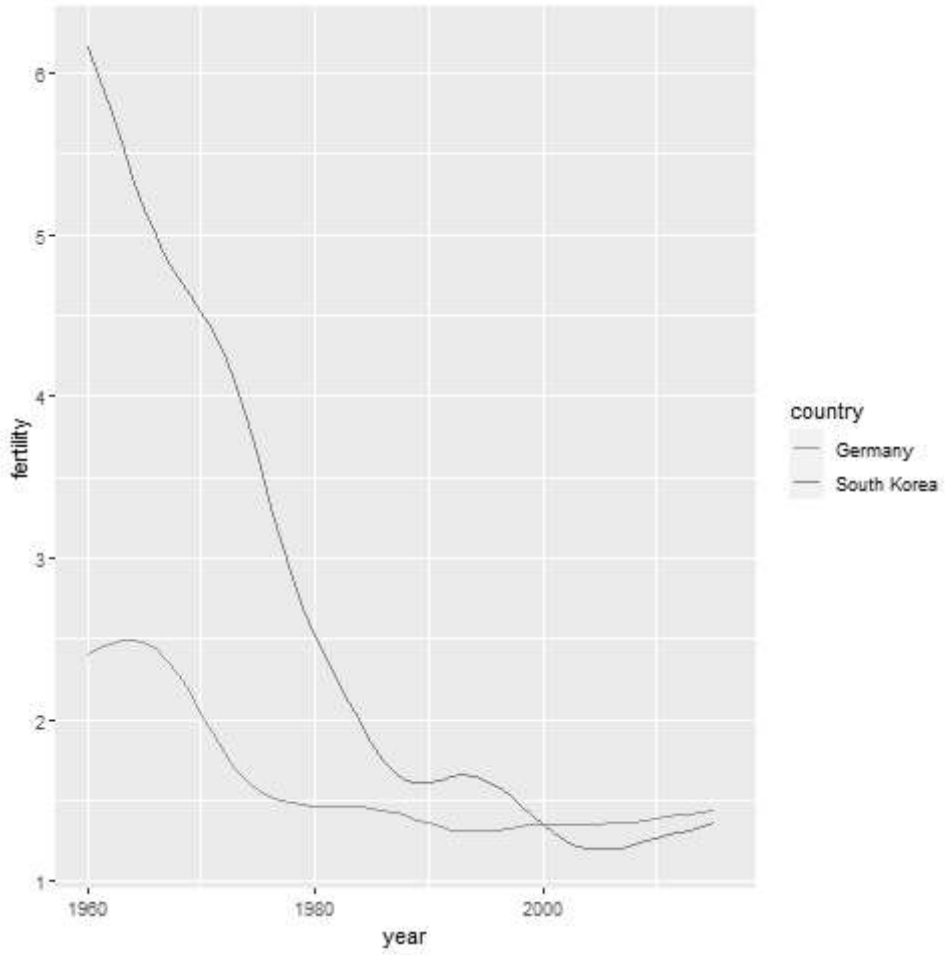
```
countries <- c("South Korea", "Germany")
gapminder %>%
  filter(country %in% countries & !is.na(fertility)) %>%
  ggplot(aes(x = year,y = fertility, group = country)) +
  geom_line()
```



- [ ]

  But **which line** belongs to which country? Use the `color` argument to assign different colors to different countries. Useful: the `color` argument to `aes` automatically groups the data (so `group` is implied).

```
countries <- c("South Korea", "Germany")
gapminder %>%
  filter(country %in% countries & !is.na(fertility)) %>%
  ggplot(aes(x = year,y = fertility, color = country)) +
  geom_line()
```



- At this point, aren't you curious what happened in South Korea between 1960 and 1990? Check it here.

1.6 Fertility rates over time for two countries: text labels

- For trend plots, labeling is clearer than legend, as long as there are not too many lines present.
- [ ]

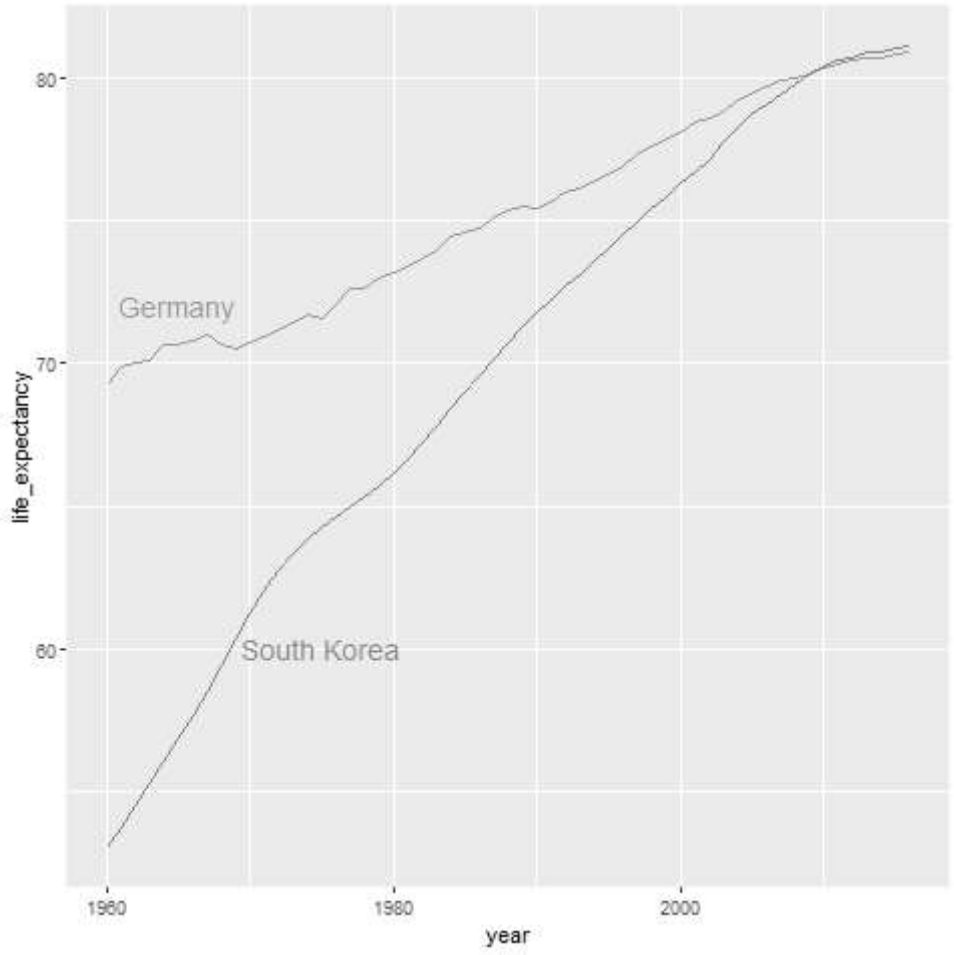
  The geometry `geom_text()` is responsible for text labels. Make a **labelled** time series plot of `year` vs. `life_expectancy` for both countries, `"South Korea"` and `"Germany"`.

  1. define a data frame called `labels` with three elements:
     - a vector of countries: `country = countries`
     - a pair (x,y) for positioning the text labels: `x = c(1975, 1965)` and `y = c(60, 72)`
  2. add a `geom_text()` layer for the `labels` data. For the mapping, use the x,y vectors and the `label` argument `country`.
  3. add a `theme` layer that removes the legend.

  You find the code below: make sure you understand it and run it.

```
labels <- data.frame(country = countries,
                     x = c(1975, 1965),
                     y = c(60,72))
gapminder %>%
  filter(country %in% countries ) %>%
```

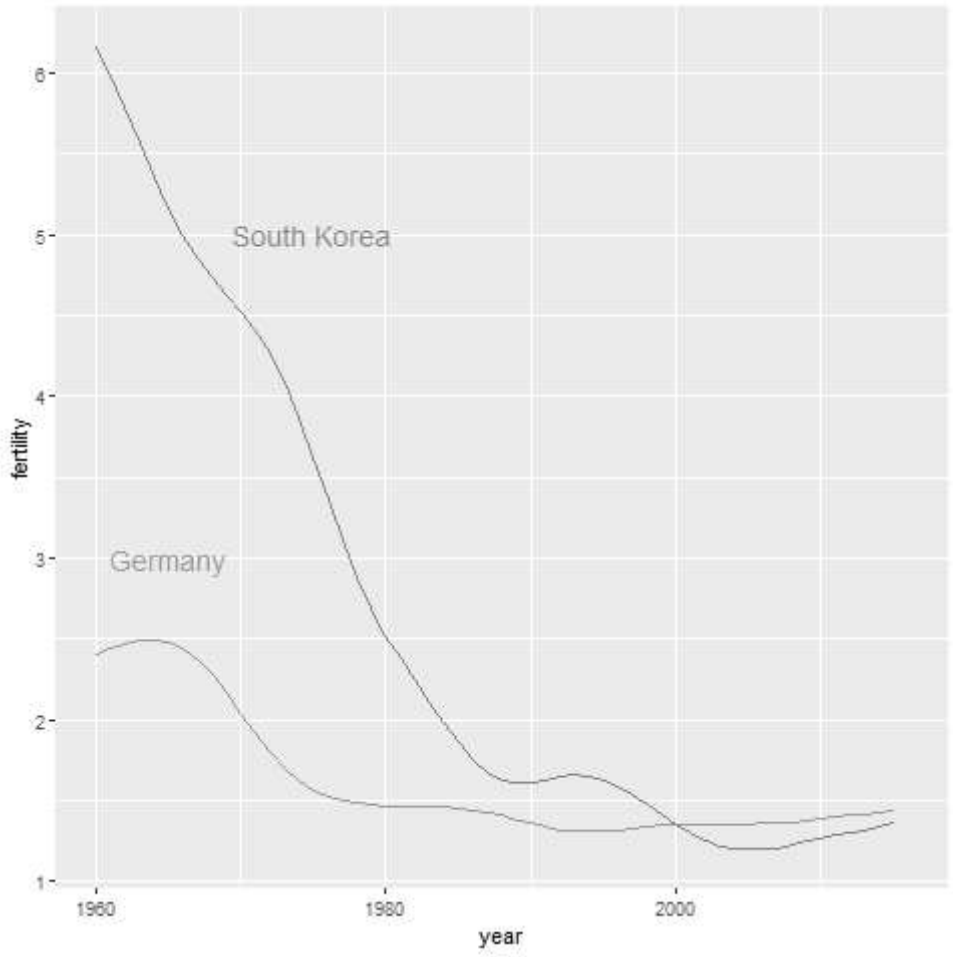
```
ggplot(aes(x = year,y = life_expectancy, color = country)) +
geom_line() +
geom_text(data = labels,
          aes(x,y, label = country), size = 5) +
theme(legend.position = "none")
```



- `[ ]`

  Now do it yourself: change the plot from earlier in 1 , `fertility vs. year` from a plot with legend to a plot with labels for both countries, and remove the legend.

  Tip: look at the plot before setting the position vectors.



Footnotes:

[1] Do you remember how to check for the number of `NA` in a vector like `fertility` in the `gapminder` data frame? Think about it for a moment, then open the code chunk below.

```
sum(is.na(gapminder$fertility))
sum(is.na(gapminder$fertility))/length(gapminder$fertility) * 100
```

```
[1] 187
[1] 1.773352
```

Author: Marcus Birkenkrahe
Created: 2022-04-06 Wed 16:40