itns Chapter 04

Peter Baumgartner

2019-05-29

Contents

1 Personal remark and setup **1**
- 1.1 Global options . . . 1
- 1.2 Installing and loading R packages . . . 2
- 1.3 Theme adaption for the graphic display with `ggplot2` . . . 2

2 End-of-Chapter Exercises **3**
- 2.1 Find z scores . . . 3
- 2.2 Find percentage of better students . . . 3
- 2.3 Calculation of SE . . . 4
  - 2.3.1 Estimation 1 . . . 4
  - 2.3.2 Estimation 2 . . . 4
  - 2.3.3 Calculation . . . 4
  - 2.3.4 Is the sample size sufficient? . . . 4
- 2.4 Nursing home and random sampling . . . 5
- 2.5 Skewed distributions . . . 5
- 2.6 Warning sign for skewed variables . . . 7

3 Additional challenge: **7**
- 3.1 Getting data from a table on a web page . . . 7
  - 3.1.1 Get table data with web scraping . . . 7
  - 3.1.2 Copy table data into a Excel sheet . . . 10
- 3.2 Tidy data . . . 11
  - 3.2.1 Tidy data frame from web scraping . . . 11
  - 3.2.2 Tidy data from Excel sheet . . . 13
- 3.3 Display distribution for age at time of death . . . 17

1 Personal remark and setup

In this chapter are most of the end-of-chapter exercises not calculation but reflections. I have almost always used the original text for questions and answers. To indicate these quotes are the text passages written in italics and marked with bar on the left margin.

As there are only few R calculations in this chapter I have added an additional challenge: Drawing a distribution for age at time of death. Data are taken from a html table on the web. This results in three different exercises:

1. Getting data via web scraping and cleaning the data frame
2. Getting data via Excel file and cleaning the data frame
3. Displaying the distribution with the program package `ggplot2`

1.1 Global options

```
### setting up working environment
### for details see: https://yihui.name/knitr/options/
knitr::opts_chunk$set(
```

```
        echo = T,
        message = T,
        error = T,
        warning = T,
        comment = '##',
        highlight = T,
        prompt = T,
        strip.white = T,
        tidy = T
        )
```

1.2 Installing and loading R packages

```
> ### https://www.tidyverse.org/
> if (!require("tidyverse")) {
+     install.packages("tidyverse", repos = "http://cran.wu.ac.at/")
+     library(tidyverse)
+ }
```

```
## Loading required package: tidyverse
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
## -- Attaching packages ---------------------------------------------------------------------------- t
```

```
## v ggplot2 3.1.1     v purrr   0.3.2
## v tibble  2.1.1     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------- tidyvers
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
> ### above command installed and loaded the core tidyverse packages: ggplot2:
> ### data visualisation tibble: a modern take on data frames tidyr: data
> ### tidying readr: data import (csv, tsv, fwf) purrr: functional R programming
> ### dplyr: data (frame) manipulation stringr: string manipulation forcats:
> ### working with categorial varialbes
```

1.3 Theme adaption for the graphic display with ggplot2

```
> my_theme <- theme_light() + theme(plot.title = element_text(size = 10, face = "bold",
+     hjust = 0.5))
> theme(plot.background = element_rect(color = NA, fill = NA)) + theme(plot.margin = margin(1,
+     0, 0, 0, unit = "cm"))
```

```
## List of 2
##  $ plot.background:List of 5
##   ..$ fill         : logi NA
##   ..$ colour       : logi NA
##   ..$ size         : NULL
```

```
##   ..$ linetype     : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ plot.margin    : 'margin' num [1:4] 1cm 0cm 0cm 0cm
##   ..- attr(*, "valid.unit")= int 1
##   ..- attr(*, "unit")= chr "cm"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

2 End-of-Chapter Exercises

2.1 Find z scores

For a standardized exam of statistics skill, scores are normally distributed: = 80, = 5. Find each student's z score:

a. Student 1: X = 80
b. Student 2: X = 90
c. Student 3: X = 75
d. Student 4: X = 95

The formula is $z = (X -)/$.

```
> (80 - 80)/5  # a.
```

```
## [1] 0
```

```
> (90 - 80)/5  # b.
```

```
## [1] 2
```

```
> (75 - 80)/5  # c.
```

```
## [1] -1
```

```
> (95 - 80)/5  # d.
```

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

a. $z = 0$
b. $z = 2$
c. $z = -1$
d. $z = 3$

2.2 Find percentage of better students

For each student in Exercise 1, use R to find what percent of students did better. (Assume X is a continuous variable.)

I am using the `pnorm` command. You can get an explanation by using the help command `help(pnorm)` or `help(Normal)`:

```
> help(Normal)
> (1 - pnorm(0)) * 100
```

```
## [1] 50
```

```
> (1 - pnorm(2)) * 100
```

```
## [1] 2.275013
```

```
> (1 - pnorm(-1)) * 100
```

```
## [1] 84.13447
```

```
> (1 - pnorm(3)) * 100
```

```
## [1] 0.1349898
```

Percent better: a. 50%; b. 2.28%; c. 84.1%; d. 0.1%.

2.3 Calculation of SE

Gabriela and Sylvia are working as a team for their university's residential life program. They are both tasked with surveying students about their satisfaction with the dormitories. Today, Gabriela has managed to survey 25 students; Sylvia has managed to survey 36 students. The satisfaction scale they are using has a range from 1 to 20 and is known from previous surveys to have = 5.

2.3.1 Estimation 1

No mathematics, just think: which sample will have the smaller SE: the one collected by Gabriela or the one collected by Sylvia?

Sylvia's sample will have the smaller SE because she has collected a larger sample.

2.3.2 Estimation 2

When the two combine their data, will this shrink the SE or grow it?

Combining the two samples will yield a smaller SE.

2.3.3 Calculation

Now calculate the SE for Gabriela's sample, for Sylvia's sample, and for the two samples combined.

The formula is $SE = /\sqrt{N}$.

```
> 5/sqrt(25)
```

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

```
> 5/sqrt(36)
```

```
## [1] 0.8333333
```

```
> 5/sqrt(25 + 36)
```

```
## [1] 0.6401844
```

For Gabriela, SE = 1; For Sylvia, SE = 0.83; Combined, SE = 0.64.

2.3.4 Is the sample size sufficient?

How big a sample size is needed? Based on the combined SE you obtained, does it seem like the residential life program should send Gabriela and Sylvia out to collect more data? Why or why not? This is a judgment call, but you should be able to make relevant comments. Consider not only the SE but the range of the measurement.

What sample size is sufficient is a judgment call, which we'll discuss further in Chapter 10. For now we can note that the combined data set provides SE = 0.64, meaning that many repeated samples would give sample mean satisfaction scores that would bounce around (i.e., form a mean heap) with standard deviation of 0.64. Given that satisfaction has a theoretical range from 1 to 20, this suggests that any one sample mean will provide a moderately precise estimate, reasonably close to the population mean. This analysis suggests we have sufficient data, although collecting more would of course most likely give us a better estimate.

2.4 Nursing home and random sampling

Rebecca works at a nursing home. She'd like to study emotional intelligence amongst the seniors at the facility (her population of interest is all the seniors living at the facility). Which of these would represent random sampling for her study?

a) Rebecca will wait in the lobby and approach any senior who randomly passes by.
b) Rebecca will wait in the lobby. As a senior passes by she will flip a coin. If the coin lands heads she will ask the senior to be in the study, otherwise she will not.
c) Rebecca will obtain a list of all the residents in the nursing home. She will randomly select 10% of the residents on this list; those selected will be asked to be part of the study.
d) Rebecca will obtain a list of all the residents in the nursing home. She will randomly select 1% of the residents on this list; those selected will be asked to be part of the study.

c and d represent random sampling because both give each member of the population an equal chance to be in the study, and members of the sample are selected independently

2.5 Skewed distributions

Sampling distributions are not always normally distributed, especially when the variable measured is highly skewed. Below are some variables that tend to have strong skew. a) In real estate, home prices tend to be skewed. In which direction? Why might this be? b) Scores on easy tests tend to be skewed. In which direction? Why might this be? c) Age of death tends to be skewed. In which direction? Why might this be? d) Number of children in a family tends to be skewed. In which direction? Why might this be?

ad a) Home prices tend to be positively skewed (longer tail to the right), because there is a lower boundary of zero, but in effect no maximum—typically a few houses have extremely high prices. These form the long upper tail of the distribution.

ad b) Scores on an easy test tend to be negatively skewed (longer tail to the left). If the test is very easy, most scores will be piled up near the maximum, but there can still be a tail to the left representing a few students who scored poorly.

ad c) Age at time of death tends to be negatively skewed (longer tail to the left). Death can strike at any time ( ), leading to a long lower tail; however, many people (in wealthy countries) die at around 70–85 years old, and no one lives forever, so the distribution is truncated at the upper end.

Searching for "distribution of age at death", or similar, will find you graphs showing this strong negative skew.

What follows are two examples for this negatively skewed distributions of age at time of death:

ad d) Number of children in a family tends to be positively skewed (longer tail to the right) because 0 is a firm minimum, and then scores extend upward from there, with many families having, say, 1–4 children and a small number of families having many children.

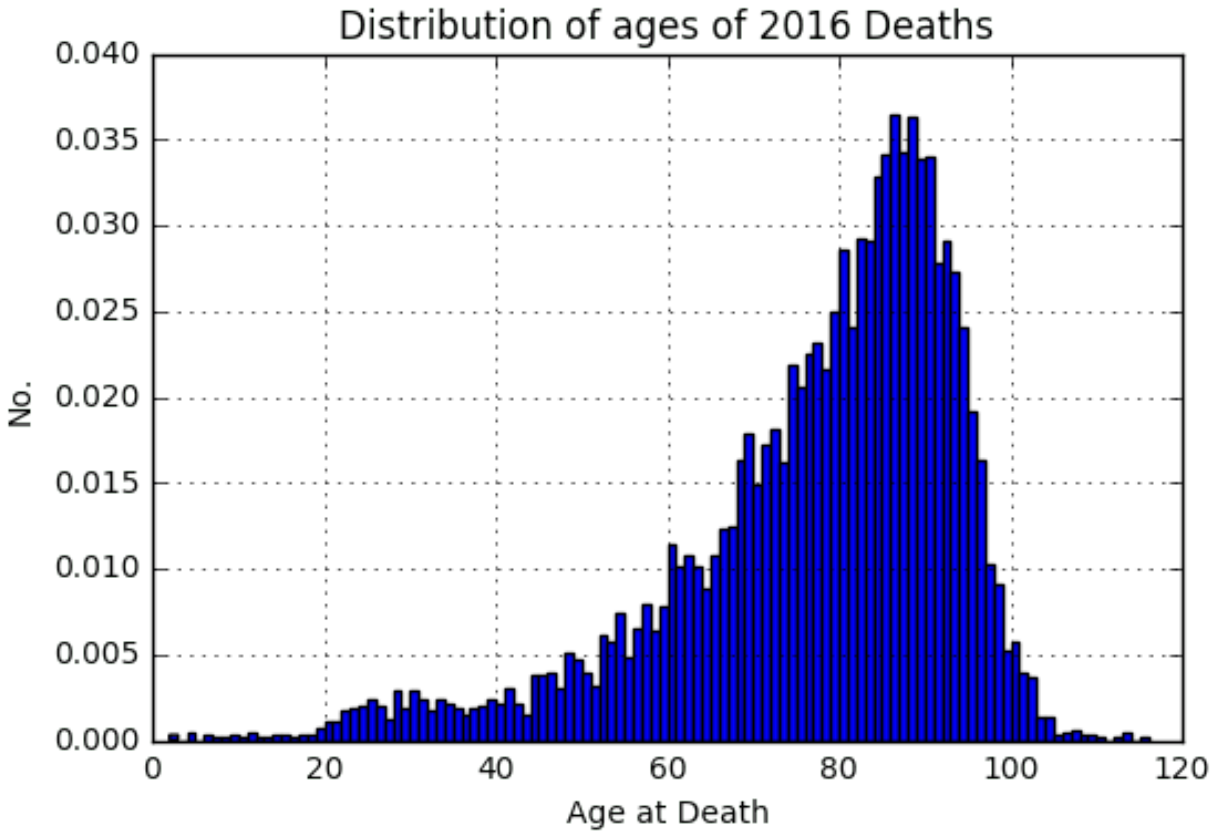


Figure 1: **Figure 1:** Celebrities death recorded by wikipedia: https://medium.com/@chris.wallace/was-2016-an-especially-bad-year-for-celebrity-deaths-40030e611f4f

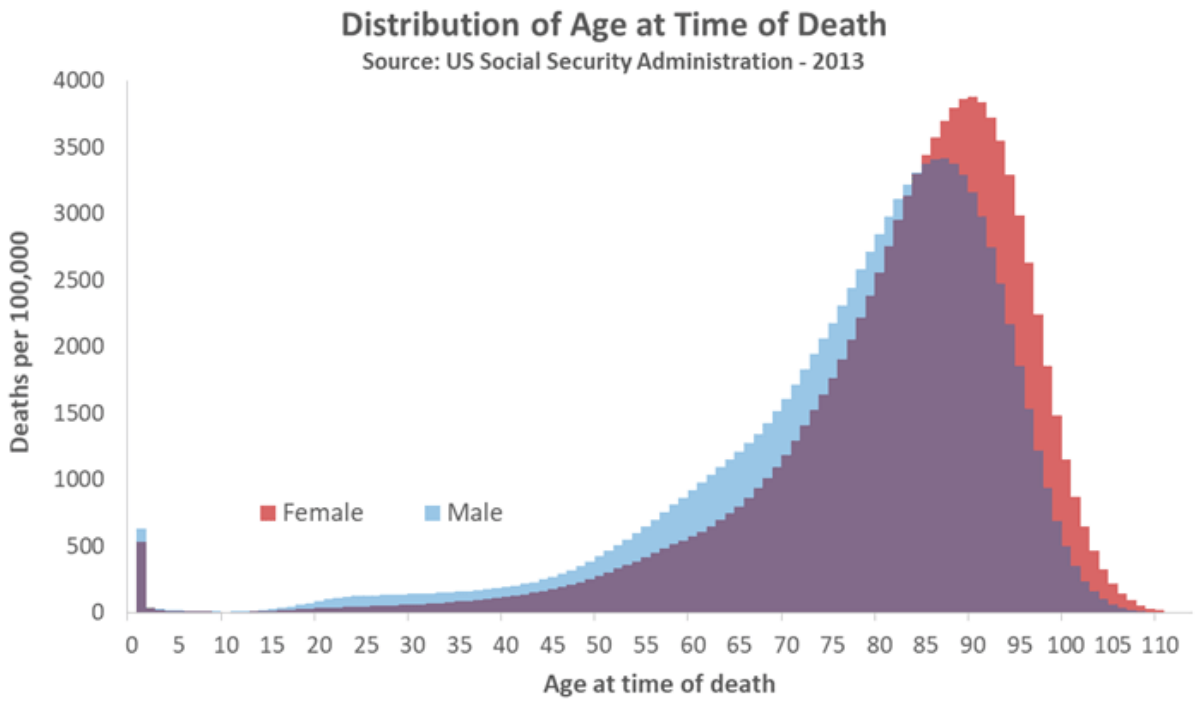


Figure 2: **Figure 2:** US-Distribution 2013 of age at time of death: https://www.quora.com/What-is-the-most-common-age-to-die-in-America

2.6 Warning sign for skewed variables

Based on the previous exercise, what is a caution or warning sign that a variable will be highly skewed?

Anything that limits, filters, selects, or caps scores on the high or low end can lead to skew. Selection is not the only thing that can produce skew, but any time your participants have been subjected to some type of selection process you should be alert to the possibility of skew in the variables used to make the selection (and in any related variables).

Also, if the mean and median differ by more than a small amount, most likely there is skew, with the mean being "pulled" in the direction of the longer tail.

3 Additional challenge:

Both graphs above are calculated from data. The first one from data on wikipedia using python, the second one used data from a life table of the US social security administration.

This opens up two questions for exercises:

1. How to get data from websites and not especially prepared excel sheets?
2. How to draw these above distributions?

3.1 Getting data from a table on a web page

3.1.1 Get table data with web scraping

To get data from web pages is called **web scraping**. You will find with a search term line "R web scraping" many articles, tutorial and videos how to do it. Here I am going to use a blog post by Cory Nissen.

We are going to use the R package `rvest` to write an appropriate script.

Look at the US period life table from 2016: How to get the necessary data for the updates graph of figure 2?

```
> # install/load package `rvest`
> if (!require("rvest"))
+         {install.packages("rvest", repos = 'http://cran.wu.ac.at/')
+         library(rvest)}
```

```
## Loading required package: rvest
```

```
## Loading required package: xml2
```

```
##
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:purrr':
##
##     pluck
```

```
## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
> # store the web url of the page with the table you are interested in
> url <- "https://www.ssa.gov/oact/STATS/table4c6.html"
> life_table_2016 <- url %>%
+   read_html() %>% # from package xml2
+   ## 1) go to webpage via google chrome
+   ## 2) set cursor on start of the desired table data
+   ## 3) right clicked and chose "inspect element"
+   ## 4) look for the appropiate line <table ...> in the inspector (typically some lines above)
+   ## 5) select this <table ...> line in the inspector
+   ## 6) right click it and select "Copy -> Copy XPath"
+   ## 7) include the copied XPath into next line of the R script
+   html_nodes(xpath='//*[@id="content"]/section[2]/div/div[3]/table') %>%
+   html_table(fill = TRUE)
> life_table_2016 <- life_table_2016[[1]]
```

To clarify how to get the correct XPath data compare the following screenshots:

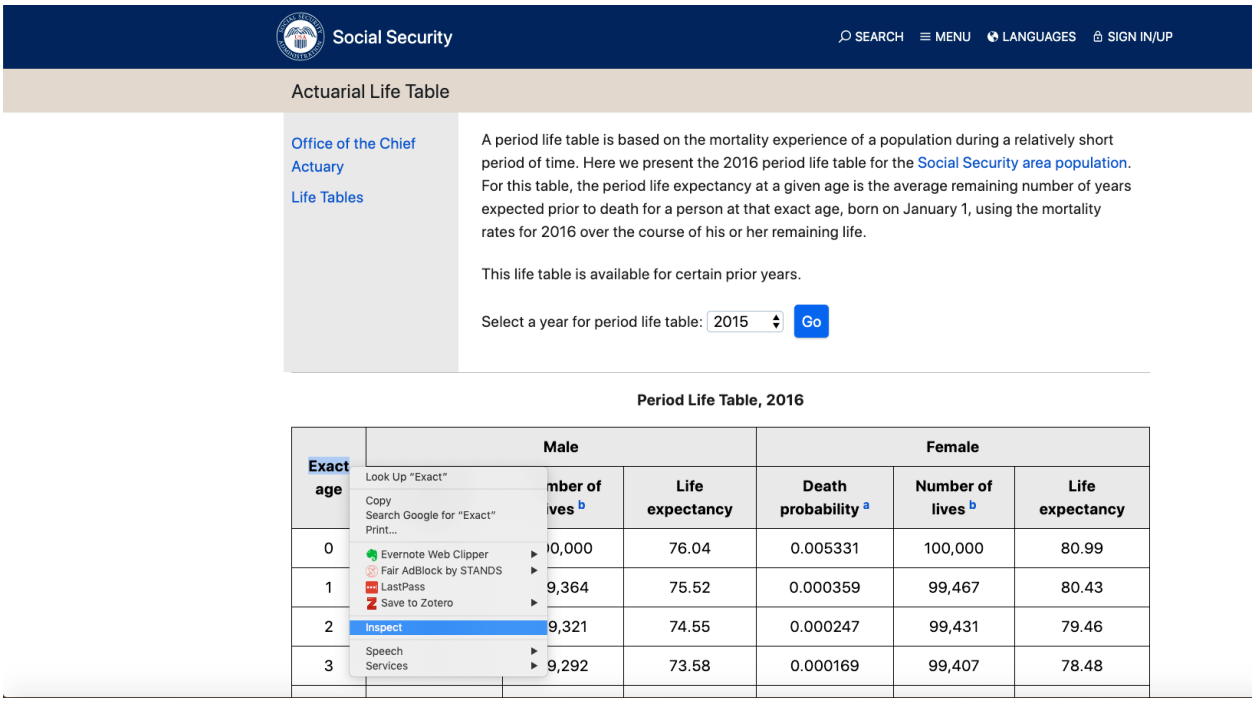


Figure 3: **Screenshot 1:** Steps 1-3 of web scraping: Set cursor on start of the desired table data, click right mouse button to get the context menue and chose 'inspect element'

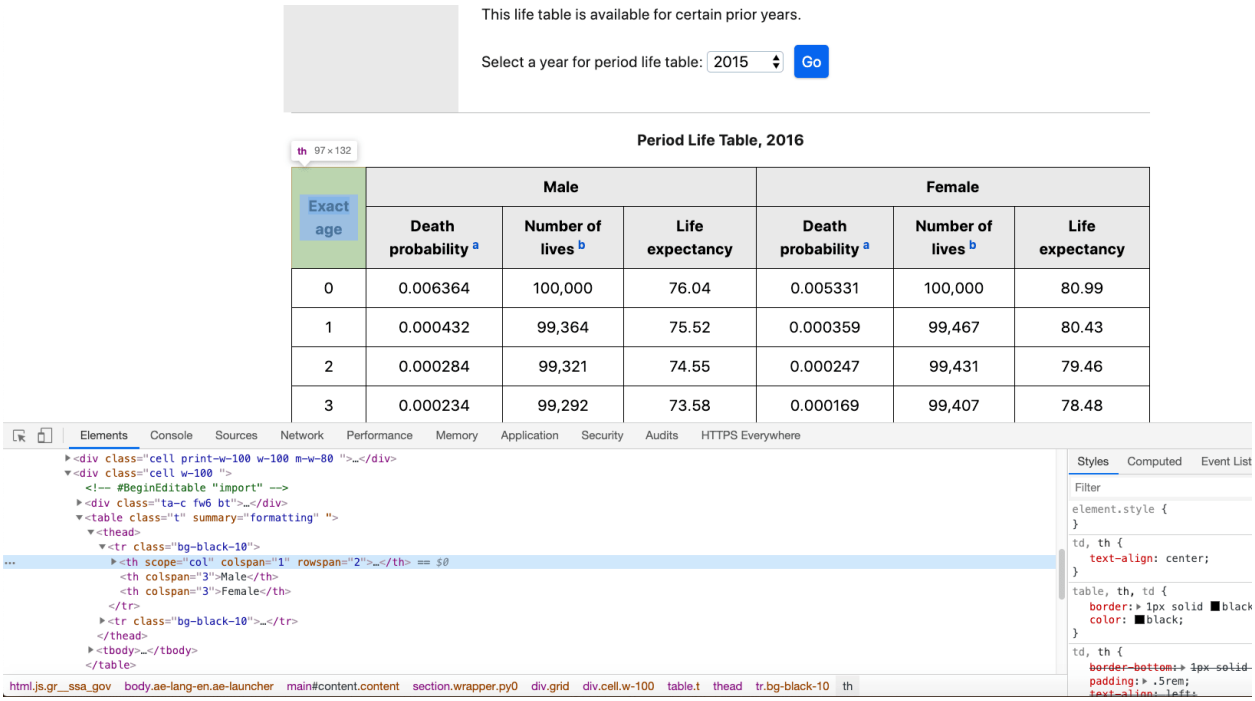
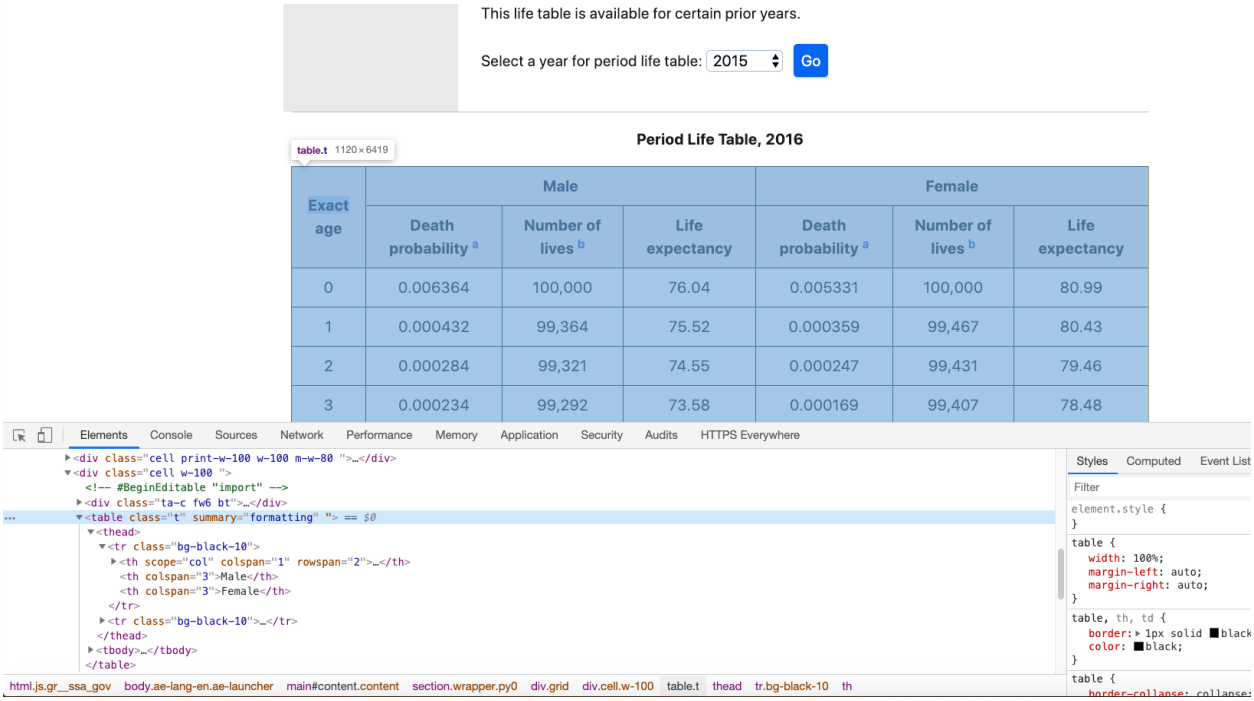
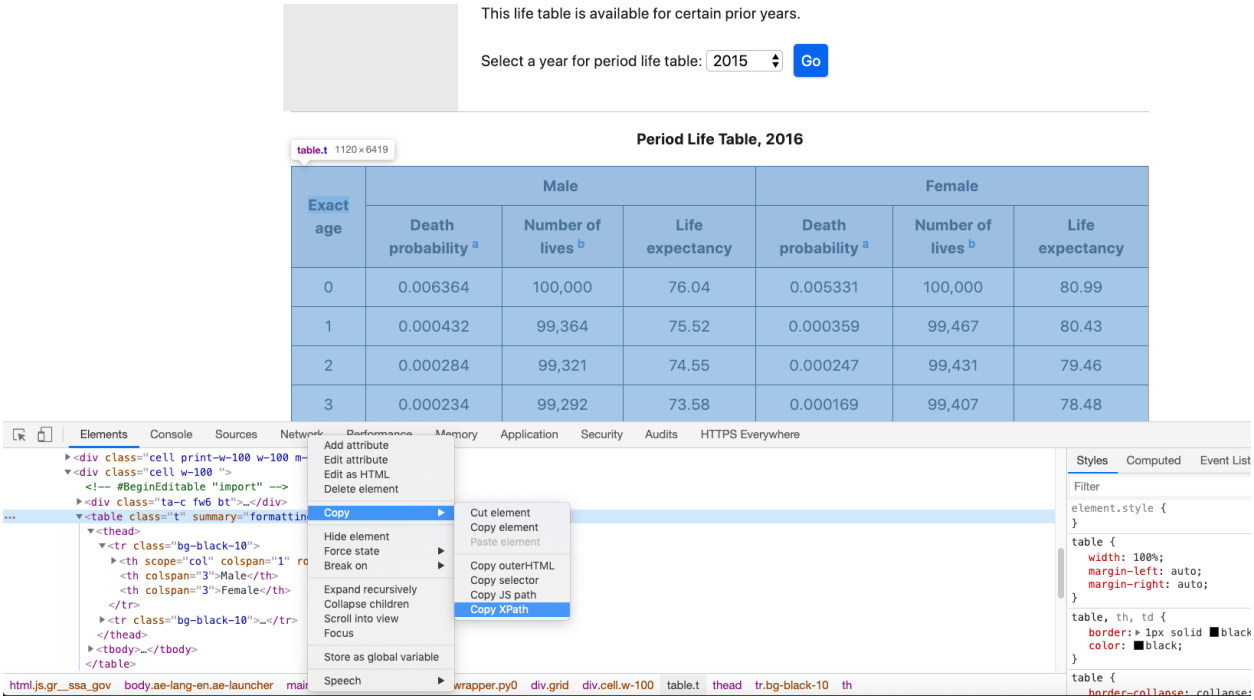


Figure 4: **Screenshot 2:** Step 4 of web scraping: look for the appropiate line <table …> in the inspector (typically some lines above)





3.1.2 Copy table data into a Excel sheet

Another simple way is to

1. select the table on the web page (without header and footnotes)
2. copy it into the clipboard
3. fire up Excel
4. set the cursor of the first cell of an empty Excel sheet
5. paste the content of the clipboard into the Excel sheet
6. save the data as a .csv file in your appropriate RStudio project folder

R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under the terms of the GNU General Public License versions 2 or 3. For more information about these matters see http://www.gnu.org/licenses/.

Note: In order to save the data in the correct format you must use the (american) number format with ',' as grouping character and '.' as the decimal character.

3.2 Tidy data

3.2.1 Tidy data frame from web scraping

Looking at the start and tail of the table we see that we need to make the following changes:

1) We only need column 1, 3 and 6 for displaying the age distribution at time of the deaths.
2) We need to delete the first and last row as it includes textual header information.
3) We need to convert the character columns into columns containing numbers.
4) We need to calculate the number of death from the survival numbers.

```
> # Looking at start and end of table
> head(life_table_2016)
```

```
##   Exact age                       Male                     Male
## 1 Exact age Death\r\n    probability a Number of\r\n   lives b
## 2         0                    0.006364                 100,000
## 3         1                    0.000432                  99,364
## 4         2                    0.000284                  99,321
## 5         3                    0.000234                  99,292
## 6         4                    0.000170                  99,269
##              Male                     Female                   Female
## 1 Life expectancy Death\r\n   probability a Number of\r\n   lives b
## 2           76.04                    0.005331                 100,000
## 3           75.52                    0.000359                  99,467
## 4           74.55                    0.000247                  99,431
## 5           73.58                    0.000169                  99,407
## 6           72.59                    0.000155                  99,390
##                   Female
## 1 Life\r\n   expectancy
## 2                 80.99
## 3                 80.43
## 4                 79.46
## 5                 78.48
## 6                 77.49
```

```
> tail(life_table_2016)
```

```
##
## 117
## 118
## 119
## 120
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
##
## 117
## 118
## 119
## 120
```

```
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
##
## 117
## 118
## 119
## 120
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
##
## 117
## 118
## 119
## 120
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
##
## 117
## 118
## 119
## 120
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
##
## 117
## 118
## 119
## 120
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
##
## 117
## 118
## 119
## 120
## 121
## 122 a Probability of dying within one year.\r\n   b Number of survivors out of 100,000 born alive.\r
```

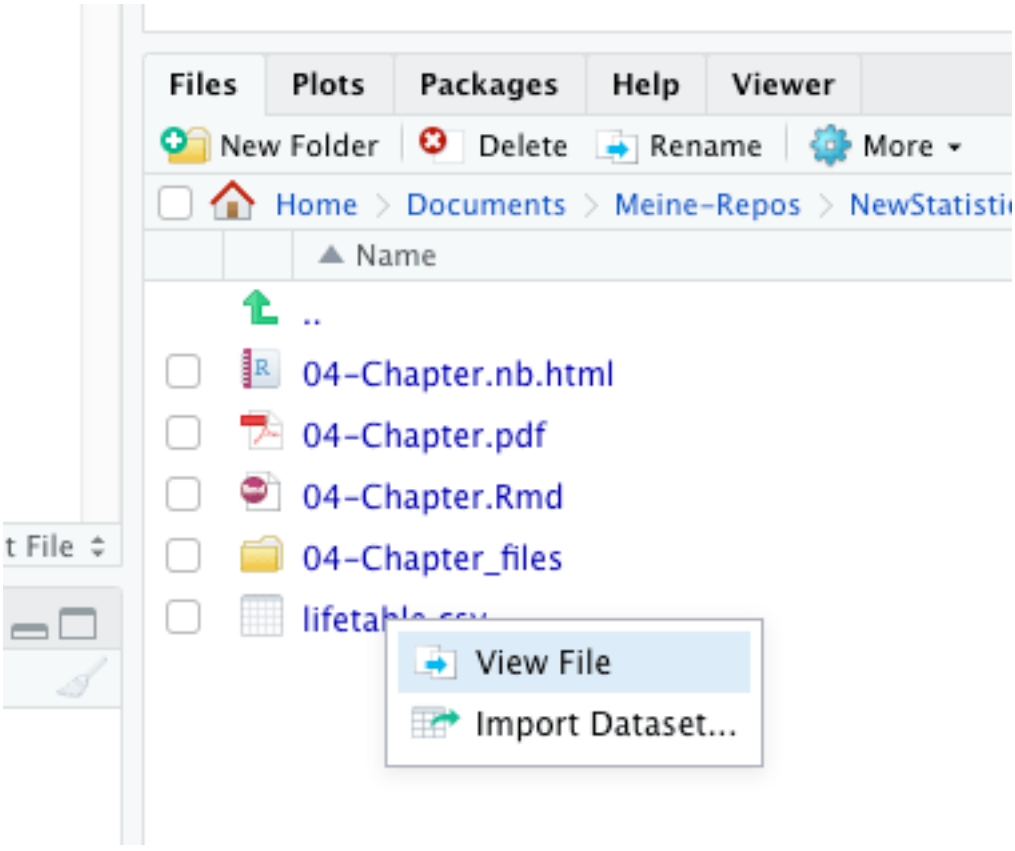
```
> # We only need column 1, 3 and 6 and have to delete the last (text) line
> life_table_2016_v2 <- life_table_2016[c(-1, -nrow(life_table_2016)), c(1, 3,
+     6)]
>
> # We need to convert the character columns into columns containing numbers
> # In the columns with big number we have additionally to delete all commas
> # from the strings
> life_table_2016_v2$`Exact age` <- as.numeric(life_table_2016_v2$`Exact age`)
> life_table_2016_v2$Male <- as.numeric(gsub(",", "", life_table_2016_v2$Male))
> life_table_2016_v2$Female <- as.numeric(gsub(",", "", life_table_2016_v2$Female))
>
>
> # We calculate the difference between rows
> life_table_2016_v3 <- abs(tail(life_table_2016_v2[, -1], -1) - head(life_table_2016_v2[,
+     -1], -1))
> names(life_table_2016_v3)[1] <- "MaleDeath"
```

```
> names(life_table_2016_v3)[2] <- "FemaleDeath"
>
> # To bind the two data frames we need the same number of rows
> firstLine <- data.frame(0, 0)
> names(firstLine)[1] <- "MaleDeath"
> names(firstLine)[2] <- "FemaleDeath"
>
> life_table_2016_v3 <- rbind(firstLine, life_table_2016_v3)
> lifetable_final <- cbind(life_table_2016_v2, life_table_2016_v3)
>
> # For easier access change variable name 'Exact age' to 'age'
> names(lifetable_final)[1] <- "age"
```

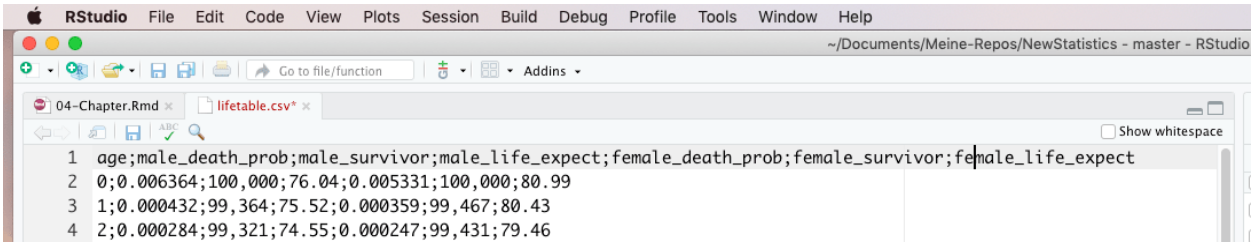
3.2.2 Tidy data from Excel sheet

We need to inspect and tidy our .csv file. Follow the procedure below:

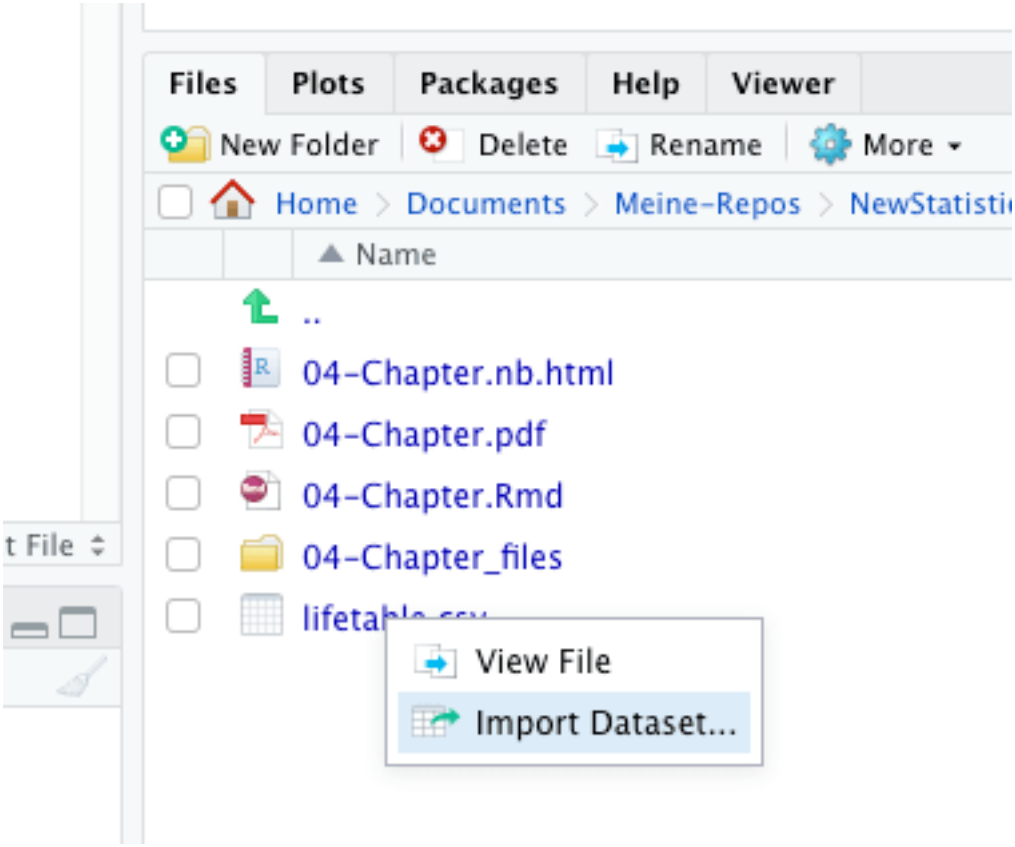
1) Click in RStudio project folder the .csv file name and select "View File" from the drop down menu which appears under your cursor



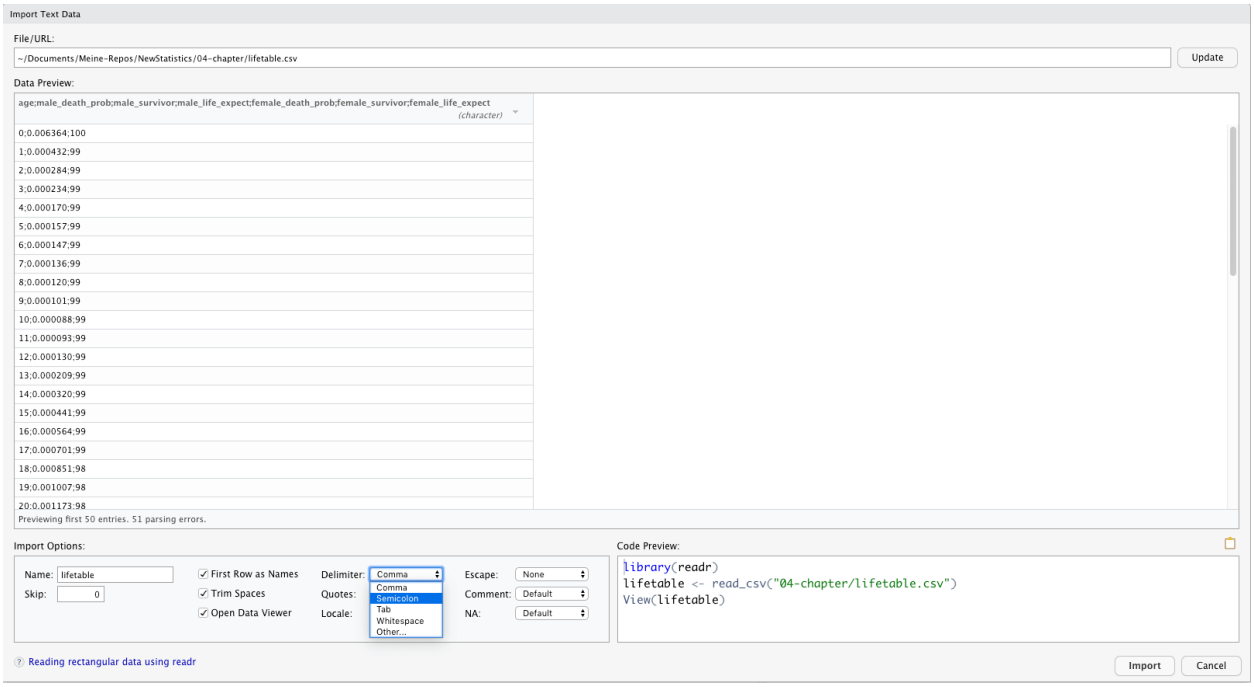
2) Insert a new line (row) at the top of the file and add column names as I did. Save the file.



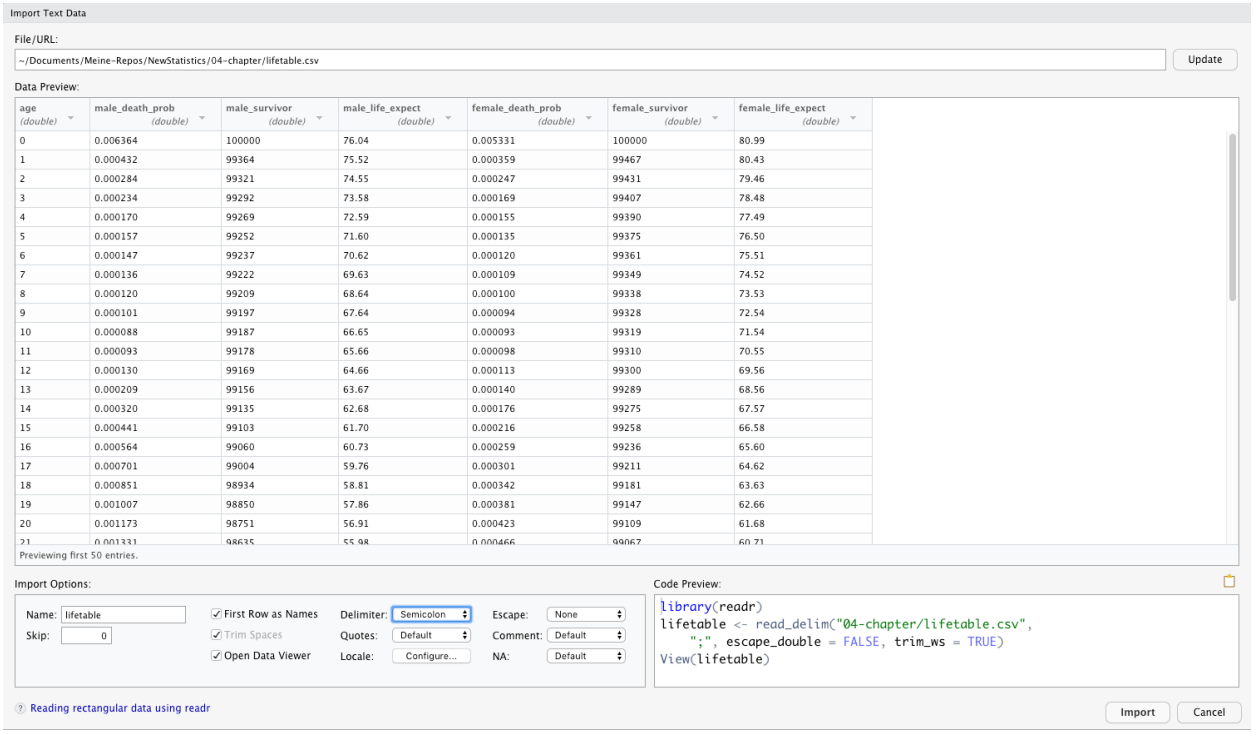
3) Click another time of the now changed .csv file name in your RStudio project folder: The same drop down menu opens up as before. This time select "Import Dataset…"



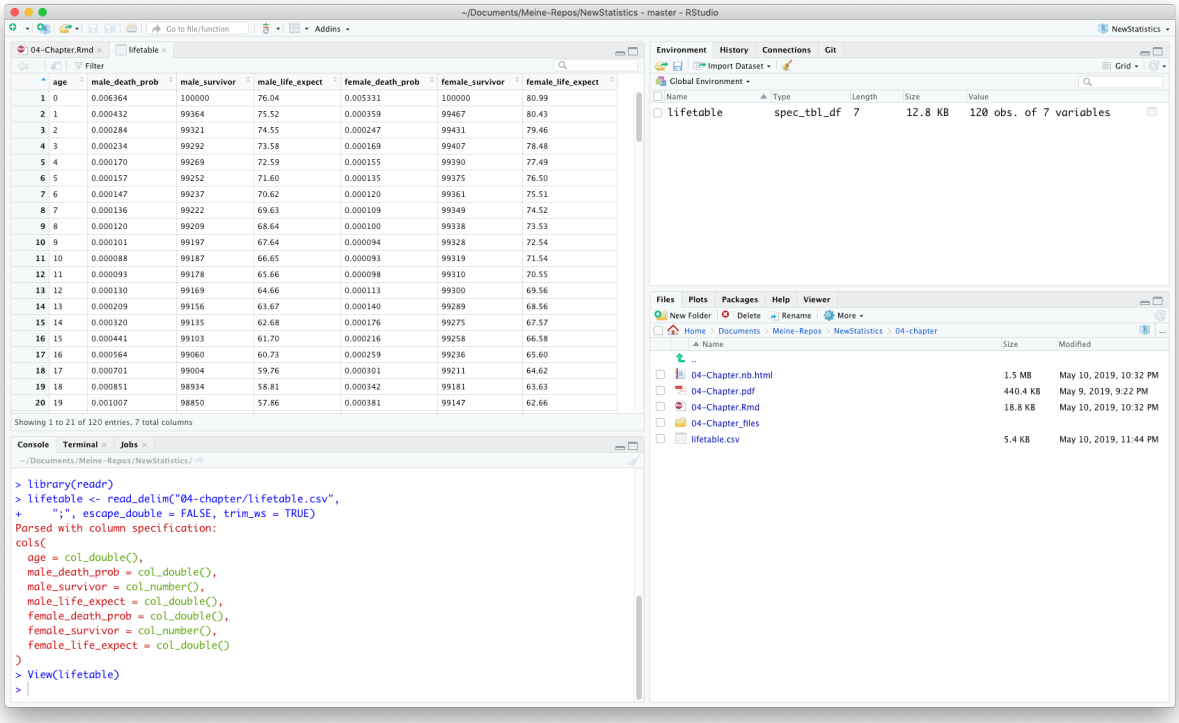
4) A overlay window to import text data appears. But the format of the data are not correct. The reason is that we have to change the delimiter from comma to semicolon.



5) Now you can see the table in the correct format. In the window in the right corner you can also see the R code which generated after you click the button “Import”. You can use the next time this code as a template for the automatic import via R script.



6) You can now inspect the data in a new tab in your code window. In the console window you can also check for the applied source code and the generated messages. You will see how the package `readr` has the data parsed and estimated the column specification.




```
> # load .csv file into variable
> lifetable <- read_delim("../04-chapter/lifetable.csv", ";", escape_double = FALSE,
+     trim_ws = TRUE)
```

```
## Parsed with column specification:
## cols(
##   age = col_double(),
##   male_death_prob = col_double(),
##   male_survivor = col_number(),
##   male_life_expect = col_double(),
##   female_death_prob = col_double(),
##   female_survivor = col_number(),
##   female_life_expect = col_double()
## )
```

```
> # We only need column 1, 3 and 6
> lifetable_v2 <- lifetable[, c(1, 3, 6)]
>
> # We calculate the difference between rows
> lifetable_v3 <- abs(tail(lifetable_v2[, -1], -1) - head(lifetable_v2[, -1],
+     -1))
> names(lifetable_v3)[1] <- "MaleDeath"
> names(lifetable_v3)[2] <- "FemaleDeath"
>
> # To bind the two data frames we need the same number of rows
> firstLine <- data.frame(0, 0)
> names(firstLine)[1] <- "MaleDeath"
> names(firstLine)[2] <- "FemaleDeath"
>
> lifetable_v3 <- rbind(firstLine, lifetable_v3)
```

```
> lifetable_final <- cbind(lifetable_v2, lifetable_v3)
```

3.3 Display distribution for age at time of death

```
> df <- reshape2::melt(lifetable_final[, c(1, 4, 5)], id.vars = "age", variable.name = "Gender")
> ggplot(df, aes(age, value)) + geom_line(aes(colour = Gender)) + labs(x = "Age of death in years",
+     y = "Death per 100,000") + scale_color_discrete(labels = c("Male", "Female")) +
+     theme(legend.position = c(0.9, 0.8))
```