Chapter 21 Exercise 21.5

James Harrington

10/21/2021

21.5 Exercises 1. Run the following command to define the co2_wide object:

```
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse
1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

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

library(dslabs)
co2_wide <- data.frame(matrix(co2, ncol = 12, byrow = TRUE)) %>%
  setNames(1:12) %>%
  mutate(year = as.character(1959:1997))
```

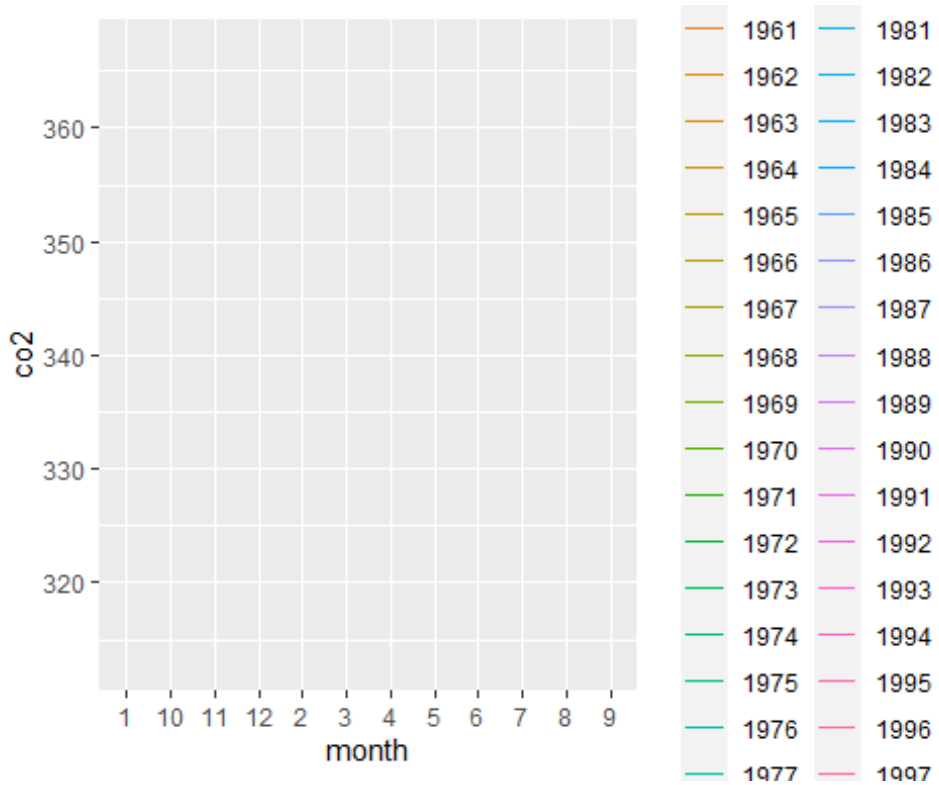
Use the pivot_longer function to wrangle this into a tidy dataset. Call the column with the CO2 measurements co2 and call the month column month. Call the resulting object co2_tidy.

```
co2_tidy <- gather(co2_wide, key = "month",value = "co2", 1:12)# "Key =" is
for the variables while "value=" is for measurement
```

2. Plot CO2 versus month with a different curve for each year using this code:

```
co2_tidy %>% ggplot(aes(month, co2, color = year)) + geom_line()

## geom_path: Each group consists of only one observation. Do you need to
adjust
## the group aesthetic?
```



If the expected plot is not made, it is probably because co2_tidy$month is not numeric:

```
class(co2_tidy$month)
```

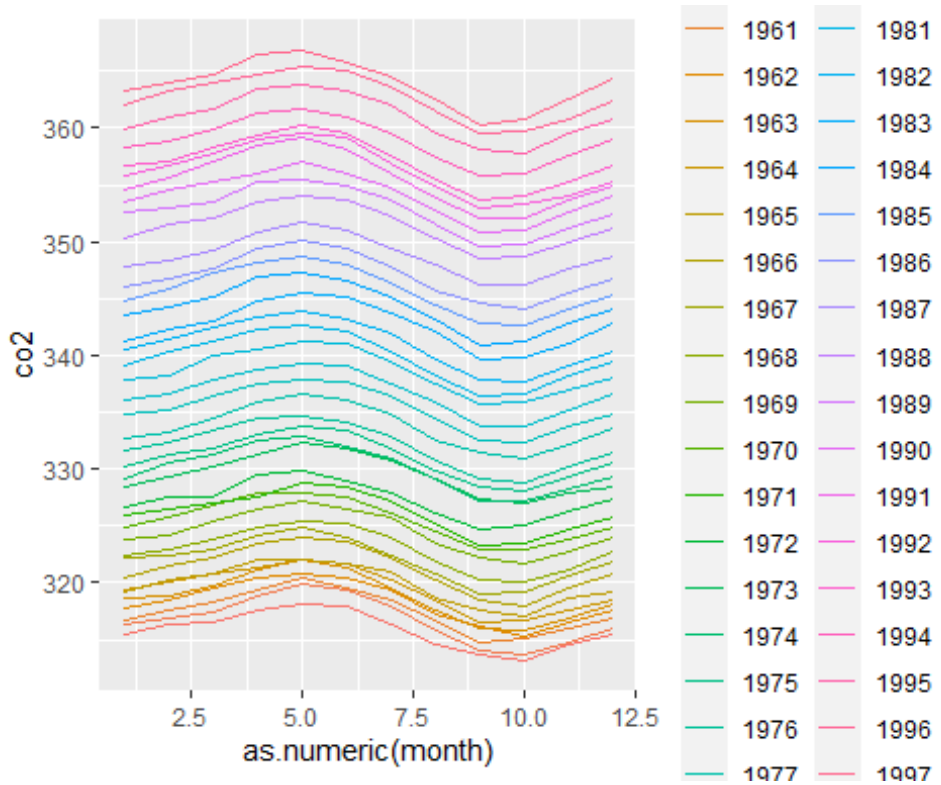
```
## [1] "character"
```

Rewrite your code to make sure the month column is numeric. Then make the plot.

```
class(co2_tidy$month)= "Numeric" #using the class function and setting it equal to numeric changes the class of the function to numeric
class(co2_tidy$month)#No, actually dont do this because it will not change it to numeric in the plot. Just use as.numeric with in the ggplot function to change month's class to numeric
```

```
## [1] "Numeric"
```

```
co2_tidy %>% ggplot(aes(as.numeric(month), co2, color = year)) + geom_line()
#add as.numeric() to change month from character to numeric
```



3. What do we learn from this plot?

```
#b)CO2 measures are higher in the summer and the yearly average increased
from 1959 to 1997.
```

4.Now load the admissions data set, which contains admission information for men and women across six majors and keep only the admitted percentage column:

separate(col= gender, into= c("men","women"),sep = "_",)

```
library(dplyr)#select function comes from the dplyr package
data(admissions)
dat.admissions <- admissions %>% select(-applicants)
```

If we think of an observation as a major, and that each observation has two variables (men admitted percentage and women admitted percentage) then this is not tidy. Use the pivot_wider function to wrangle into tidy shape: one row for each major.

```
dat.admissions <- admissions %>%
  pivot_wider(names_from = gender, values_from = admitted, major)
```

rlang::last_error()

5. Now we will try a more advanced wrangling challenge. We want to wrangle the admissions data so that for each major we have 4 observations: admitted_men, admitted_women, applicants_men and applicants_women. The trick we perform here is actually quite common: first use pivot_longer to generate an intermediate

data frame and then pivot_wider to obtain the tidy data we want. We will go step by step in this and the next two exercises.

Use the pivot_longer function to create a tmp data.frame with a column containing the type of observation admitted or applicants. Call the new columns name and value.

```
dat.admissions <- admissions %>%
  pivot_wider(names_from = gender, values_from = admitted, major)
```

```
library(dslabs)
library(tidyr)
data("admissions")
tmp= gather(admissions,name,value,admitted:applicants)
tmp
```

```
##    major gender       name value
## 1      A    men   admitted    62
## 2      B    men   admitted    63
## 3      C    men   admitted    37
## 4      D    men   admitted    33
## 5      E    men   admitted    28
## 6      F    men   admitted     6
## 7      A  women   admitted    82
## 8      B  women   admitted    68
## 9      C  women   admitted    34
## 10     D  women   admitted    35
## 11     E  women   admitted    24
## 12     F  women   admitted     7
## 13     A    men applicants   825
## 14     B    men applicants   560
## 15     C    men applicants   325
## 16     D    men applicants   417
## 17     E    men applicants   191
## 18     F    men applicants   373
## 19     A  women applicants   108
## 20     B  women applicants    25
## 21     C  women applicants   593
## 22     D  women applicants   375
## 23     E  women applicants   393
## 24     F  women applicants   341
```

```
tmp%>% unite("column_name", name, gender)
```

```
##    major      column_name value
## 1      A     admitted_men    62
## 2      B     admitted_men    63
## 3      C     admitted_men    37
## 4      D     admitted_men    33
## 5      E     admitted_men    28
## 6      F     admitted_men     6
## 7      A   admitted_women    82
```

```
## 8      B   admitted_women    68
## 9      C   admitted_women    34
## 10     D   admitted_women    35
## 11     E   admitted_women    24
## 12     F   admitted_women     7
## 13     A   applicants_men   825
## 14     B   applicants_men   560
## 15     C   applicants_men   325
## 16     D   applicants_men   417
## 17     E   applicants_men   191
## 18     F   applicants_men   373
## 19     A applicants_women   108
## 20     B applicants_women    25
## 21     C applicants_women   593
## 22     D applicants_women   375
## 23     E applicants_women   393
## 24     F applicants_women   341
```

7. Now use the pivot_wider function to generate the tidy data with four variables for each major.

```
library(dslabs)
library(tidyr)
data("admissions")
tmp= gather(admissions,name,value,admitted:applicants)
tmp
```

```
##    major gender       name value
## 1      A    men   admitted    62
## 2      B    men   admitted    63
## 3      C    men   admitted    37
## 4      D    men   admitted    33
## 5      E    men   admitted    28
## 6      F    men   admitted     6
## 7      A  women   admitted    82
## 8      B  women   admitted    68
## 9      C  women   admitted    34
## 10     D  women   admitted    35
## 11     E  women   admitted    24
## 12     F  women   admitted     7
## 13     A    men applicants   825
## 14     B    men applicants   560
## 15     C    men applicants   325
## 16     D    men applicants   417
## 17     E    men applicants   191
## 18     F    men applicants   373
## 19     A  women applicants   108
## 20     B  women applicants    25
## 21     C  women applicants   593
## 22     D  women applicants   375
```

```
## 23     E  women applicants   393
## 24     F  women applicants   341
```

```
tmp%>% unite("column_name", name, gender)%>%
pivot_wider(names_from = column_name, values_from = value)
```

```
## # A tibble: 6 x 5
##   major admitted_men admitted_women applicants_men applicants_women
##   <chr>        <dbl>          <dbl>          <dbl>             <dbl>
## 1 A               62             82            825               108
## 2 B               63             68            560                25
## 3 C               37             34            325               593
## 4 D               33             35            417               375
## 5 E               28             24            191               393
## 6 F                6              7            373               341
```