HW 6

March 18, 2021

1 IST 387 HW 6

[4]:
```
# Enter your name here: Connor Hanan
```

1.0.1 Attribution statement: (choose only one and delete the rest)

[5]:
```
# 1. I did this homework by myself, with help from the book and the professor.
```

Reminders of things to practice from previous weeks: Descriptive statistics: mean( ) Install a package: install.packages( ) ?command: Ask R for help with a command

This module: Data visualization is important because many people can make sense of data more easily when it is presented in graphic form. As a data scientist, you will have to present complex data to decision makers in a form that makes the data interpretable for them. From your experience with Excel and other tools, you know that there are a variety of **common data visualizations** (e.g., pie charts). How many of them can you name?

The most powerful tool for data visualization in R is called **ggplot**. Written by computer/data scientist **Hadley Wickham**, this **"graphics grammar"** tool builds visualizations in layers. This method provides immense flexibility, but takes a bit of practice to master.

1.1 Step 1: Make a copy of the data

A. Copy the **who** dataset from this URL: https://ist387.s3.us-east-2.amazonaws.com/data/who.csv into a new dataframe called **tb**. Your new dataframe, tb, contains a so-called **multivariate time series**: a sequence of measurements on 23 Tuberculosis-related (TB) variables captured repeatedly over time (1980-2013). Familiarize yourself with the nature of the 23 variables by consulting the dataset's codebook which can be found here: https://ist387.s3.us-east-2.amazonaws.com/data/TB_data_dictionary_2021-02-06.csv.

[6]:
```
library(tidyverse)
```

```
  Attaching packages                     tidyverse
1.3.0

 ggplot2 3.3.2      purrr   0.3.4
 tibble  3.0.4      dplyr   1.0.2
```

```
  tidyr   1.1.2       stringr 1.4.0
  readr   1.4.0       forcats 0.5.0

  Conflicts
tidyverse_conflicts()
  dplyr::filter() masks stats::filter()
  dplyr::lag()    masks stats::lag()
```

[34]:
```
tb <- read_csv("https://ist387.s3.us-east-2.amazonaws.com/data/who.csv")
```

```
  Column specification

cols(
  .default = col_double(),
  iso2 = col_character()
)
  Use `spec()` for the full
column specifications.
```

[9]:
```
tb
```

iso2 <chr>	year <dbl>	new_sp <dbl>	new_sp_m04 <dbl>	new_sp_m514 <dbl>	new_sp_m014 <dbl>	new_ <dbl>
AD	1989	NA	NA	NA	NA	NA
AD	1990	NA	NA	NA	NA	NA
AD	1991	NA	NA	NA	NA	NA
AD	1992	NA	NA	NA	NA	NA
AD	1993	15	NA	NA	NA	NA
AD	1994	24	NA	NA	NA	NA
AD	1996	8	NA	NA	0	0
AD	1997	17	NA	NA	0	0
AD	1998	1	NA	NA	0	0
AD	1999	4	NA	NA	0	0
AD	2000	1	NA	NA	0	0
AD	2001	3	NA	NA	0	NA
AD	2002	2	NA	NA	0	0
AD	2003	7	NA	NA	0	0
AD	2004	3	NA	NA	0	0
AD	2005	5	0	0	0	0
AD	2006	8	0	0	0	1
AD	2007	2	NA	NA	NA	NA
AD	2008	3	0	0	0	0
AE	1980	NA	NA	NA	NA	NA
AE	1981	NA	NA	NA	NA	NA
AE	1982	NA	NA	NA	NA	NA
AE	1983	NA	NA	NA	NA	NA
AE	1984	NA	NA	NA	NA	NA
AE	1985	NA	NA	NA	NA	NA
AE	1986	NA	NA	NA	NA	NA
AE	1987	NA	NA	NA	NA	NA
AE	1988	NA	NA	NA	NA	NA
AE	1989	NA	NA	NA	NA	NA
AE	1990	NA	NA	NA	NA	NA
ZM	2008	13211	NA	NA	101	1120
ZW	1980	NA	NA	NA	NA	NA
ZW	1981	NA	NA	NA	NA	NA
ZW	1982	NA	NA	NA	NA	NA
ZW	1983	NA	NA	NA	NA	NA
ZW	1984	NA	NA	NA	NA	NA
ZW	1985	NA	NA	NA	NA	NA
ZW	1986	NA	NA	NA	NA	NA
ZW	1987	NA	NA	NA	NA	NA
ZW	1988	NA	NA	NA	NA	NA
ZW	1989	NA	NA	NA	NA	NA
ZW	1990	NA	NA	NA	NA	NA
ZW	1991	NA	NA	NA	NA	NA
ZW	1992	NA	NA	NA	NA	NA
ZW	1993	5331	NA	NA	NA	NA
ZW	1994	NA	NA	NA	NA	NA
ZW	1995	8965	NA	NA	NA	NA
ZW	1996	11965	NA	NA	NA	NA
ZW	1997	14512	NA	NA	NA	NA
ZW	1998	14492	NA	NA	NA	NA

A spec_tbl_df: 5769 × 23

B. How often were these measurements taken (in other words, at what frequency were the variables measured)? Put your answer in a comment.

[ ]:
```
#measurements were taken every year for each country
```

1.2 Step 2: Clean-up the NAs with Missing Data Mitigation

A. Create a subset of **tb** containing **only the records for Canada** (**"CA" in the iso2 variable**). Save it in a new dataframe called **tbCan**. Make sure this new df has **52 observations** and **23 variables**.

[35]:
```
tb %>%
filter(iso2 == "CA") -> tbcan
#doesn't give me the right number of observations because it discards the full␣
 ↪NA rows - reluctantly abandoning dplyr for the subsetting method so I can␣
 ↪follow the steps
```

[36]:
```
tbcan <- tb[tb$iso2 == 'CA',]
```

[18]:
```
str(tbcan)
```

```
tibble [52 × 23] (S3: tbl_df/tbl/data.frame)
 $ iso2        : chr [1:52] "CA" "CA" "CA" "CA" …
 $ year        : num [1:52] 1980 1981 1982 1983 1984 …
 $ new_sp      : num [1:52] 951 803 812 771 811 791 752 668 682 652 …
 $ new_sp_m04  : num [1:52] NA NA NA NA NA NA NA NA NA NA …
 $ new_sp_m514 : num [1:52] NA NA NA NA NA NA NA NA NA NA …
 $ new_sp_m014 : num [1:52] 12 8 6 9 3 11 9 9 4 10 …
 $ new_sp_m1524: num [1:52] 54 49 52 47 44 42 58 40 43 45 …
 $ new_sp_m2534: num [1:52] 75 61 66 63 75 70 73 71 73 56 …
 $ new_sp_m3544: num [1:52] 83 64 69 62 58 59 62 60 62 60 …
 $ new_sp_m4554: num [1:52] 100 87 90 90 68 77 59 49 52 54 …
 $ new_sp_m5564: num [1:52] 108 103 91 92 83 81 73 64 68 62 …
 $ new_sp_m65  : num [1:52] 186 141 150 123 169 168 147 129 131 122 …
 $ new_sp_mu   : num [1:52] NA NA NA NA NA NA NA NA NA NA …
 $ new_sp_f04  : num [1:52] NA NA NA NA NA NA NA NA NA NA …
 $ new_sp_f514 : num [1:52] NA NA NA NA NA NA NA NA NA NA …
 $ new_sp_f014 : num [1:52] 18 6 7 11 9 5 10 8 6 6 …
 $ new_sp_f1524: num [1:52] 62 46 51 50 51 30 33 39 38 37 …
 $ new_sp_f2534: num [1:52] 51 57 57 50 59 56 54 48 56 51 …
 $ new_sp_f3544: num [1:52] 34 26 30 29 28 19 33 29 27 23 …
 $ new_sp_f4554: num [1:52] 31 28 25 24 28 28 20 17 16 24 …
 $ new_sp_f5564: num [1:52] 33 35 38 35 36 48 26 26 26 21 …
 $ new_sp_f65  : num [1:52] 104 92 80 86 100 97 95 79 80 81 …
 $ new_sp_fu   : num [1:52] NA NA NA NA NA NA NA NA NA NA …
```

A simple method for dealing with small amounts of **missing data** in a numeric variable is to **substitute the mean of the variable in place of each missing datum**. This expression

locates (and reports to the console) all the missing data elements in the variable measuring the **number of positive pulmonary smear tests for male children 0-14 years old**:

[37]:
```
tbcan$new_sp_m014[is.na(tbcan$new_sp_m014)]
```

1. <NA> 2. <NA> 3. <NA> 4. <NA> 5. <NA> 6. <NA> 7. <NA> 8. <NA> 9. <NA> 10. <NA> 11. <NA> 12. <NA> 13. <NA> 14. <NA> 15. <NA> 16. <NA> 17. <NA> 18. <NA> 19. <NA> 20. <NA> 21. <NA> 22. <NA> 23. <NA>

B. Write a comment describing how that statement works.

[23]:
```
tbcan #locate all the NA's in the denoted column, then replace it with the mean␣
↪of the rest of the data (by sorting !is.na)
```

A tibble: 52 × 23

iso2	year	new_sp	new_sp_m04	new_sp_m514	new_sp_m014	new_sp_m152
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
CA	1980	951	NA	NA	12.0000000	54
CA	1981	803	NA	NA	8.0000000	49
CA	1982	812	NA	NA	6.0000000	52
CA	1983	771	NA	NA	9.0000000	47
CA	1984	811	NA	NA	3.0000000	44
CA	1985	791	NA	NA	11.0000000	42
CA	1986	752	NA	NA	9.0000000	58
CA	1987	668	NA	NA	9.0000000	40
CA	1988	682	NA	NA	4.0000000	43
CA	1989	652	NA	NA	10.0000000	45
CA	1990	549	NA	NA	3.0000000	35
CA	1991	543	NA	NA	7.0000000	37
CA	1992	506	NA	NA	6.0000000	42
CA	1993	488	NA	NA	8.0000000	33
CA	1994	483	NA	NA	2.0000000	42
CA	1995	436	NA	NA	1.0000000	28
CA	1996	430	NA	NA	3.0000000	28
CA	1997	473	NA	NA	0.0000000	21
CA	1998	438	NA	NA	4.0000000	33
CA	1999	455	NA	NA	0.0000000	23
CA	2000	492	NA	NA	5.0000000	34
CA	2001	458	NA	NA	6.0000000	24
CA	2002	408	NA	NA	0.0000000	25
CA	2003	332	NA	NA	1.0000000	26
CA	2004	438	NA	NA	2.0000000	25
CA	2005	433	NA	NA	3.0000000	37
CA	2006	407	1	1	2.0000000	34
CA	2007	463	4	1	5.0000000	31
CA	2008	488	0	2	2.0000000	39
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA

C. Write one more statement to report missing data for the number of positive pulmonary smear tests for **female** children 0-14 years old, and a statement for the number of positive pulmonary smear tests for the **male and female citizens 65 years of age and older**, respectively.

[38]:
```
tbcan$new_sp_f014[is.na(tbcan$new_sp_f014)]
tbcan$new_sp_f65[is.na(tbcan$new_sp_f65)]
tbcan$new_sp_m65[is.na(tbcan$new_sp_m65)]
```

1. <NA> 2. <NA> 3. <NA> 4. <NA> 5. <NA> 6. <NA> 7. <NA> 8. <NA> 9. <NA> 10. <NA> 11. <NA> 12. <NA> 13. <NA> 14. <NA> 15. <NA> 16. <NA> 17. <NA> 18. <NA> 19. <NA> 20. <NA> 21. <NA> 22. <NA> 23. <NA>

1. <NA> 2. <NA> 3. <NA> 4. <NA> 5. <NA> 6. <NA> 7. <NA> 8. <NA> 9. <NA> 10. <NA> 11. <NA> 12. <NA> 13. <NA> 14. <NA> 15. <NA> 16. <NA> 17. <NA> 18. <NA> 19. <NA> 20. <NA> 21. <NA> 22. <NA> 23. <NA>

1. <NA> 2. <NA> 3. <NA> 4. <NA> 5. <NA> 6. <NA> 7. <NA> 8. <NA> 9. <NA> 10. <NA> 11. <NA> 12. <NA> 13. <NA> 14. <NA> 15. <NA> 16. <NA> 17. <NA> 18. <NA> 19. <NA> 20. <NA> 21. <NA> 22. <NA> 23. <NA>

[27]:
```
tbcan
```

A tibble: 52×23

iso2	year	new_sp	new_sp_m04	new_sp_m514	new_sp_m014	new_sp_m152
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
CA	1980	951	NA	NA	12.0000000	54
CA	1981	803	NA	NA	8.0000000	49
CA	1982	812	NA	NA	6.0000000	52
CA	1983	771	NA	NA	9.0000000	47
CA	1984	811	NA	NA	3.0000000	44
CA	1985	791	NA	NA	11.0000000	42
CA	1986	752	NA	NA	9.0000000	58
CA	1987	668	NA	NA	9.0000000	40
CA	1988	682	NA	NA	4.0000000	43
CA	1989	652	NA	NA	10.0000000	45
CA	1990	549	NA	NA	3.0000000	35
CA	1991	543	NA	NA	7.0000000	37
CA	1992	506	NA	NA	6.0000000	42
CA	1993	488	NA	NA	8.0000000	33
CA	1994	483	NA	NA	2.0000000	42
CA	1995	436	NA	NA	1.0000000	28
CA	1996	430	NA	NA	3.0000000	28
CA	1997	473	NA	NA	0.0000000	21
CA	1998	438	NA	NA	4.0000000	33
CA	1999	455	NA	NA	0.0000000	23
CA	2000	492	NA	NA	5.0000000	34
CA	2001	458	NA	NA	6.0000000	24
CA	2002	408	NA	NA	0.0000000	25
CA	2003	332	NA	NA	1.0000000	26
CA	2004	438	NA	NA	2.0000000	25
CA	2005	433	NA	NA	3.0000000	37
CA	2006	407	1	1	2.0000000	34
CA	2007	463	4	1	5.0000000	31
CA	2008	488	0	2	2.0000000	39
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA
NA	NA	NA	NA	NA	0.5576923	NA

There is an R package called **imputeTS** specifically designed to repair missing values in time series data. We will use this instead of mean substitution. The **na_interpolation()** function in this package takes advantage of a unique characteristic of time series data: **neighboring points in time can be used to "guess" about a missing value in between**.

D. Install the **imputeTS** package and use **na_interpolation( )** on the four variables whose missing values you reported on in B & C. Don't forget that you need to save the results back to the **tbCan** dataframe.

[29]:
```
#install.packages('imputeTS')
library(imputeTS)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

[45]:
```
tbcan %>%
  mutate(new_sp_f014 = na_interpolation(new_sp_f014),
         new_sp_m014 = na_interpolation(new_sp_m014),
        new_sp_f65 = na_interpolation(new_sp_f65),
        new_sp_m65 = na_interpolation(new_sp_m65)) -> tbcan
```

E. Rerun the code from B & C above to check that all missing data have been fixed.

[46]:
```
tbcan$new_sp_m014[is.na(tbcan$new_sp_f014)]
tbcan$new_sp_f014[is.na(tbcan$new_sp_f014)]
tbcan$new_sp_f65[is.na(tbcan$new_sp_f65)]
tbcan$new_sp_m65[is.na(tbcan$new_sp_m65)]
```

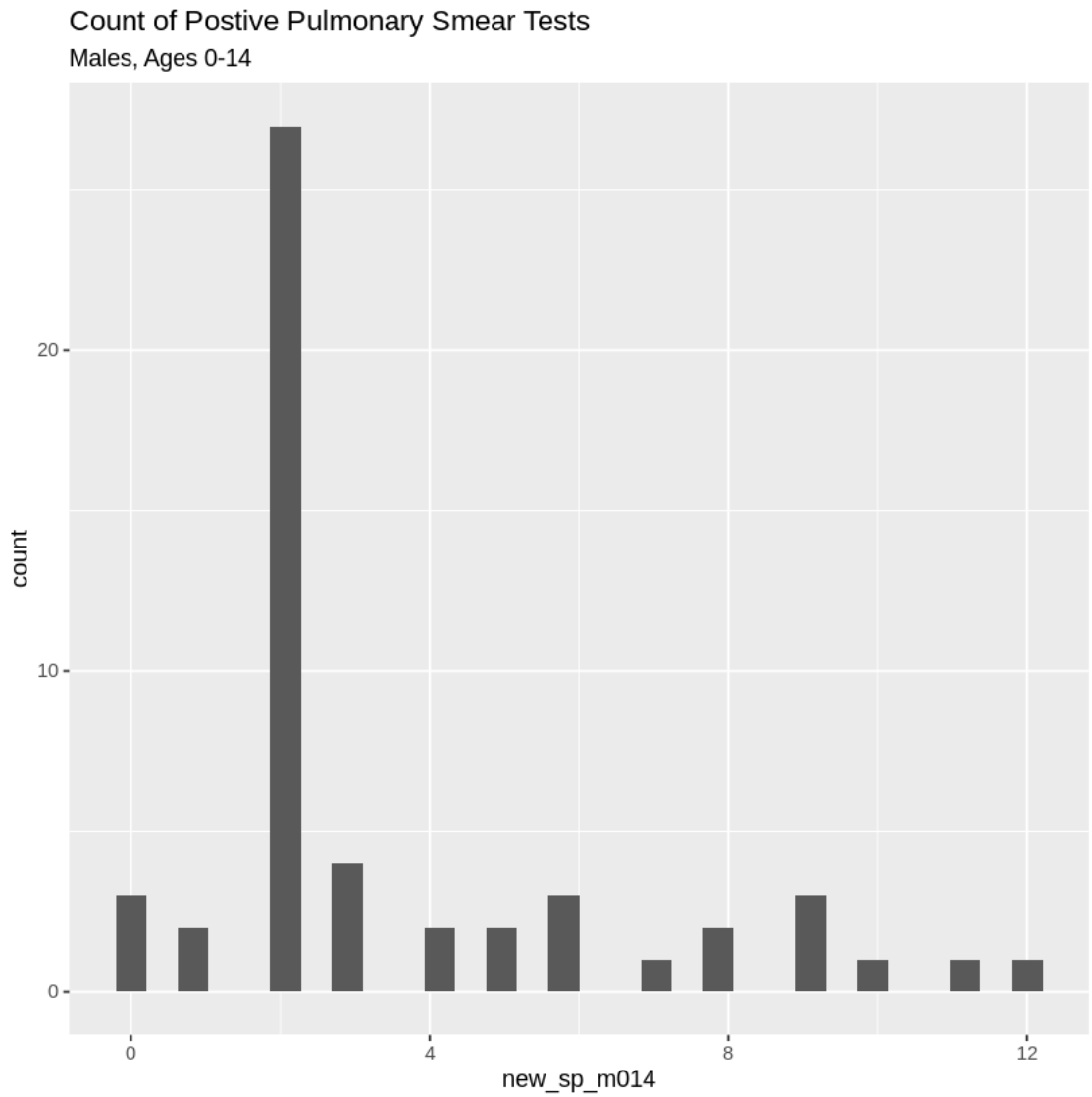
1.3 Step 3: Use ggplot to explore the distribution of each variable

Don't forget to install and library the ggplot2 package. Then: F. Create a histogram for **new_sp_m014**. Be sure to add a title and briefly describe what the histogram means in a comment.

[47]:
```
library(tidyverse)
```

[49]:
```
tbcan %>%
ggplot() +
geom_histogram(aes(new_sp_m014)) +
ggtitle("Count of Postive Pulmonary Smear Tests", subtitle = "Males, Ages 0-14")

#this histogram shows the Count of Postive Pulmonary Smear Tests in Males, Ages␣
 ↪0-14
```

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



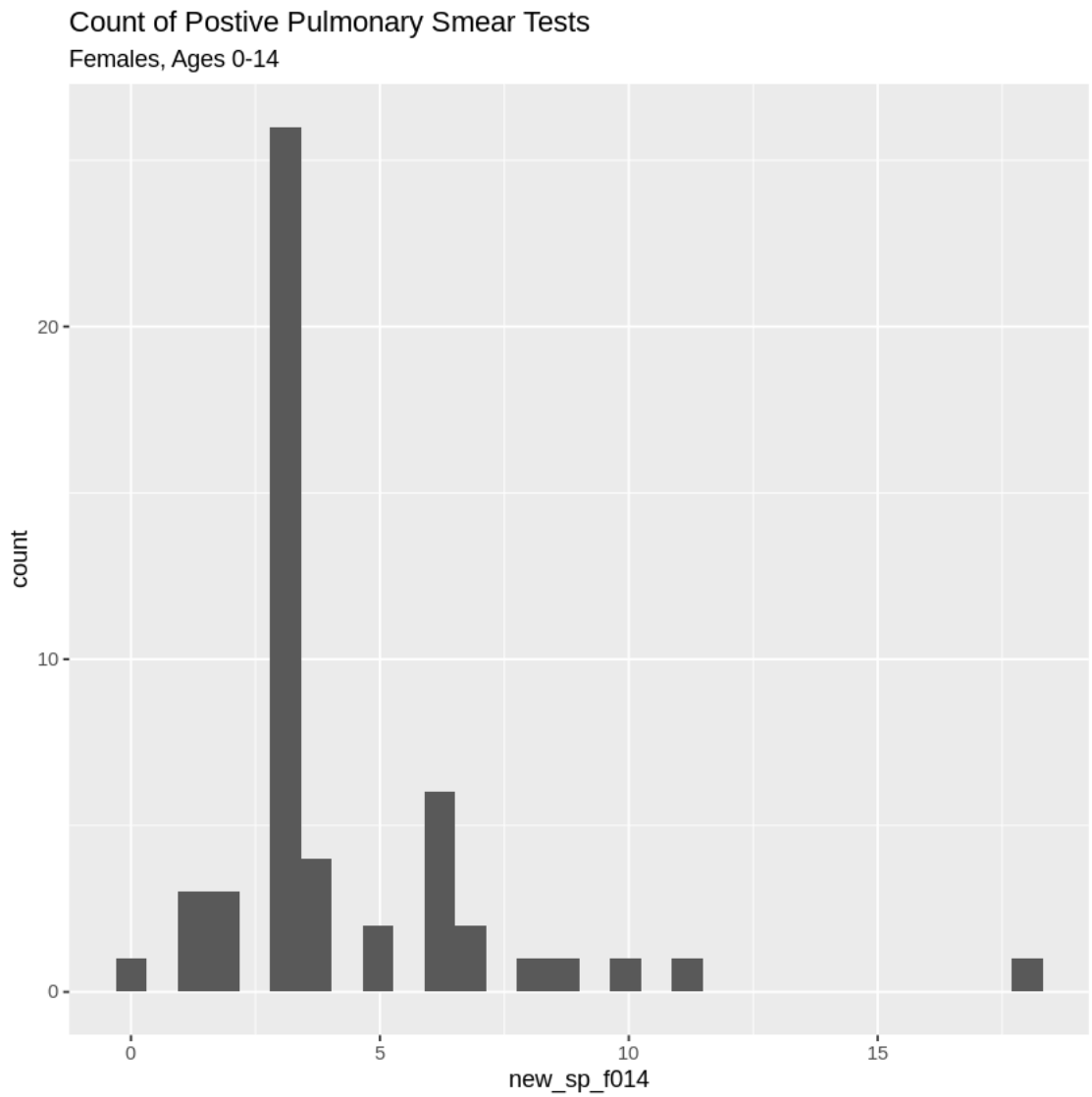
G. Create histograms of each of the other three variables with ggplot( ). Which parameter do you need to adjust to make the other histograms look right?

[51]:
```
tbcan %>%
ggplot() +
geom_histogram(aes(new_sp_f014)) +
ggtitle("Count of Postive Pulmonary Smear Tests", subtitle = "Females, Ages␣
↪0-14")

tbcan %>%
ggplot() +
geom_histogram(aes(new_sp_m65)) +
ggtitle("Count of Postive Pulmonary Smear Tests", subtitle = "Males, Ages 65+")
```

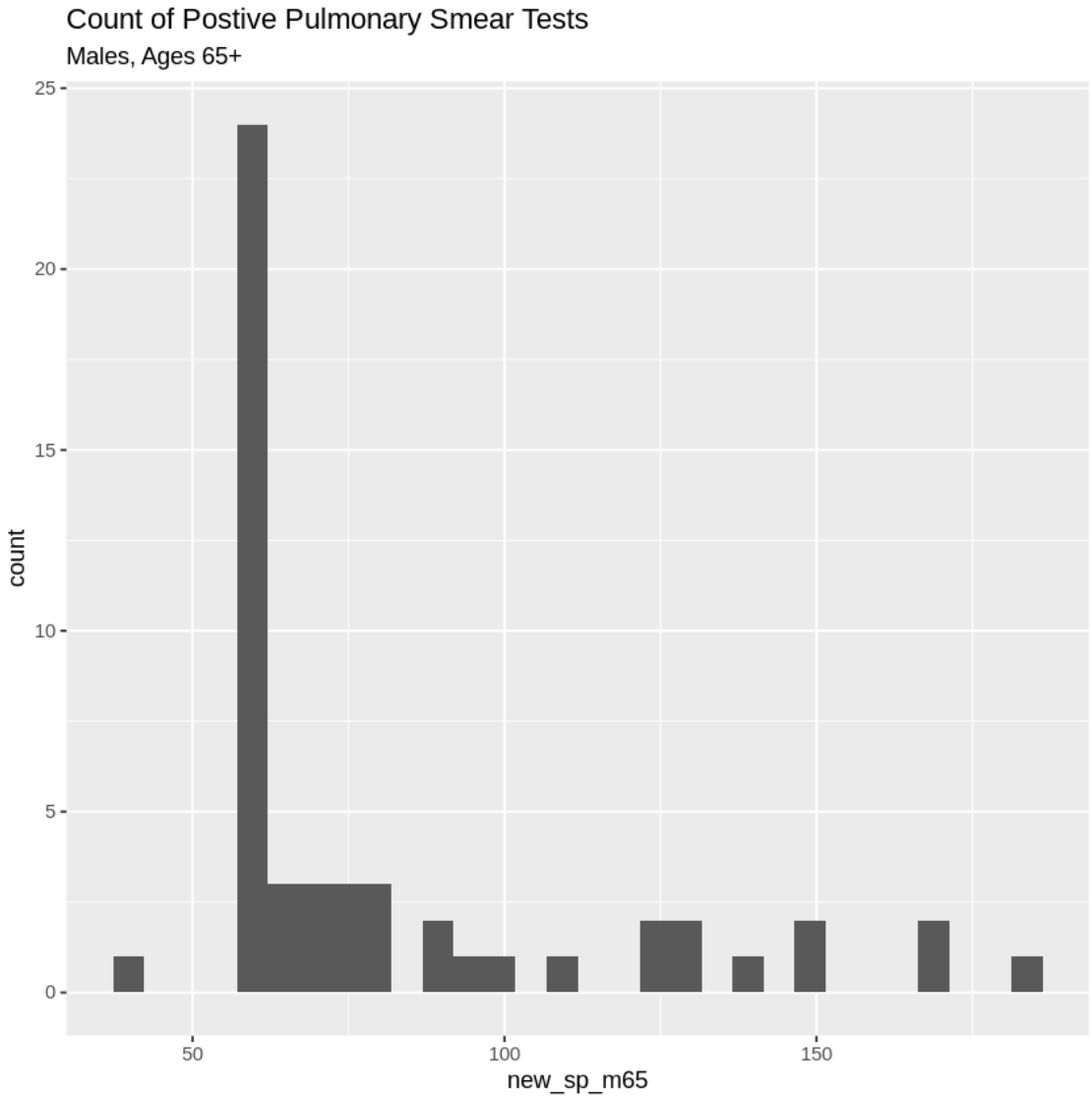
```
tbcan %>%
ggplot() +
geom_histogram(aes(new_sp_f65)) +
ggtitle("Count of Postive Pulmonary Smear Tests", subtitle = "Females, Ages␣
 ↪65+")

#you have to change the x aesthetic each time, as well as the title
```

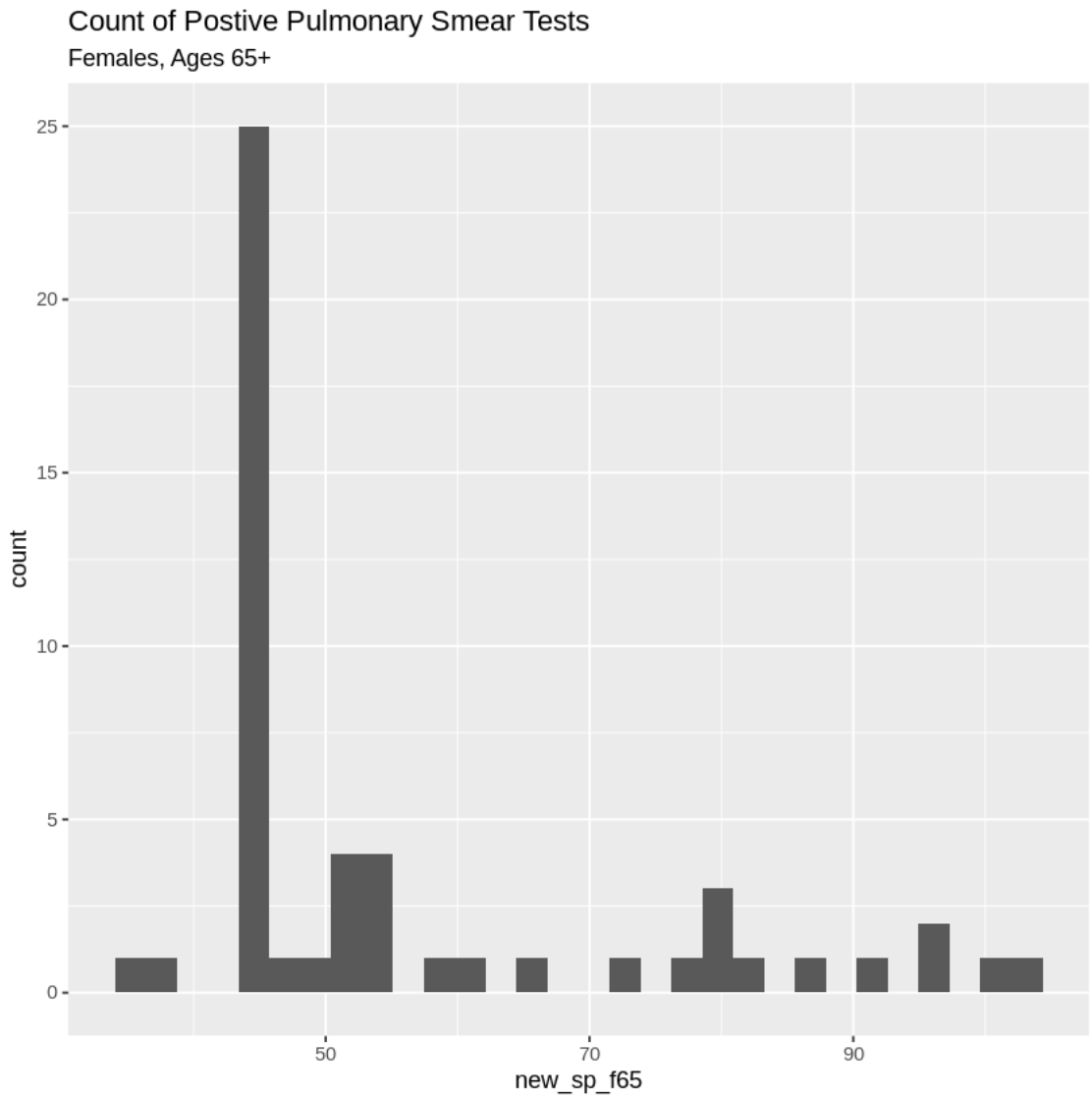
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

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



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





1.4 Step 4: Explore how the data changes over time

H. These data were collected in a period of several decades. You can thus observe changes over time with the help of a line chart. Create a **line chart**, with **year** on the X-axis and **new_sp_m014** on the Y-axis.

[58]:
```
library(scales)
```

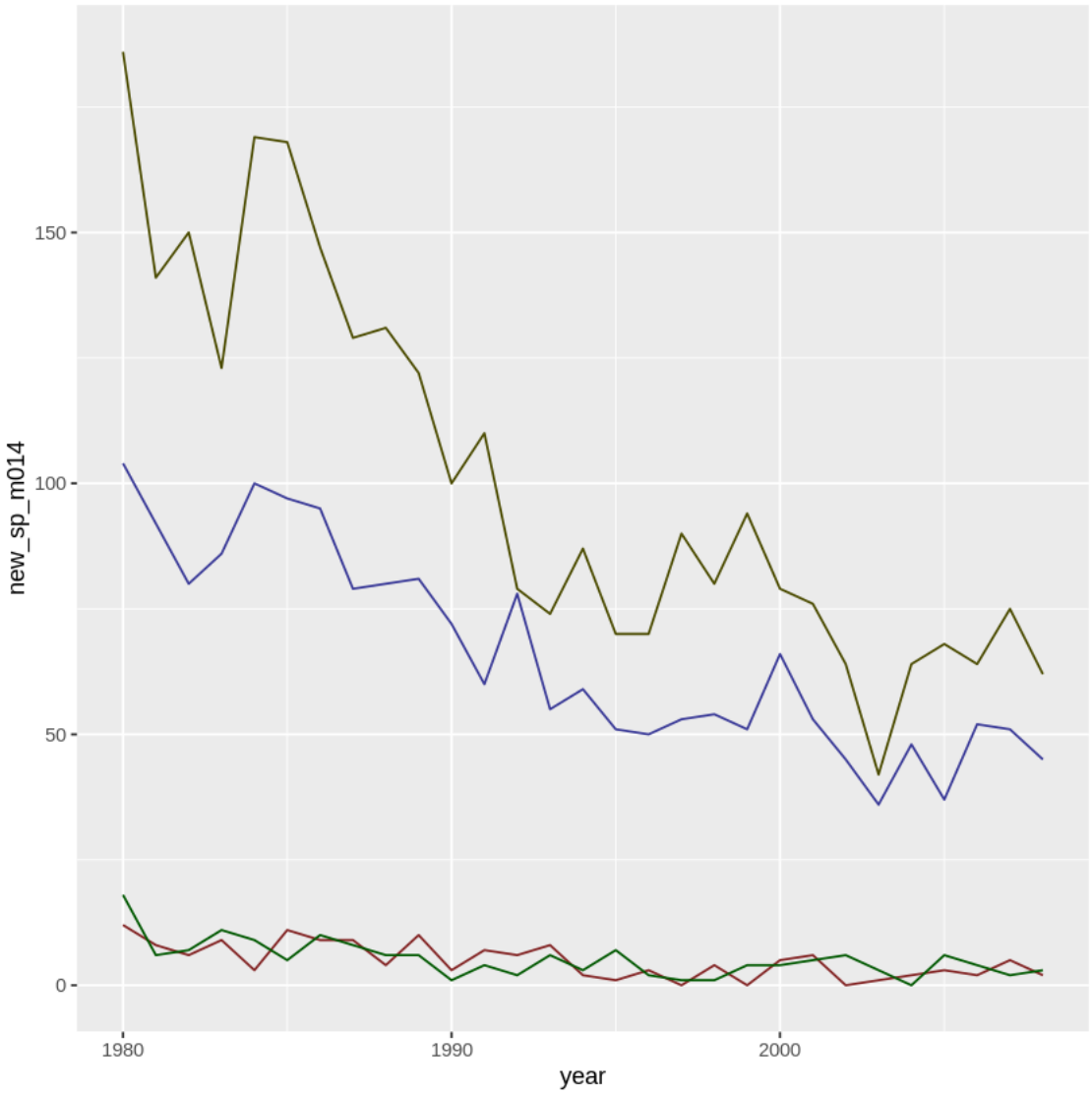
```
Attaching package: 'scales'
```

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

```
    discard


The following object is masked from ‘package:readr’:

    col_factor
```

[61]:
```
tbcan %>%
ggplot()+
geom_line(aes(year,new_sp_m014), color = muted('red'))+
geom_line(aes(year,new_sp_f014), color = muted('green'))+
geom_line(aes(year,new_sp_m65), color = muted('yellow'))+
geom_line(aes(year,new_sp_f65), color = muted('blue'))
```

```
Warning message:
"Removed 23 row(s) containing missing values (geom_path)."
Warning message:
"Removed 23 row(s) containing missing values (geom_path)."
Warning message:
"Removed 23 row(s) containing missing values (geom_path)."
Warning message:
"Removed 23 row(s) containing missing values (geom_path)."
```



I. Next, create similar graphs for each of the other three variables. Change the **color** of the line plots (any color you want).

[ ]:
```
#see above
```

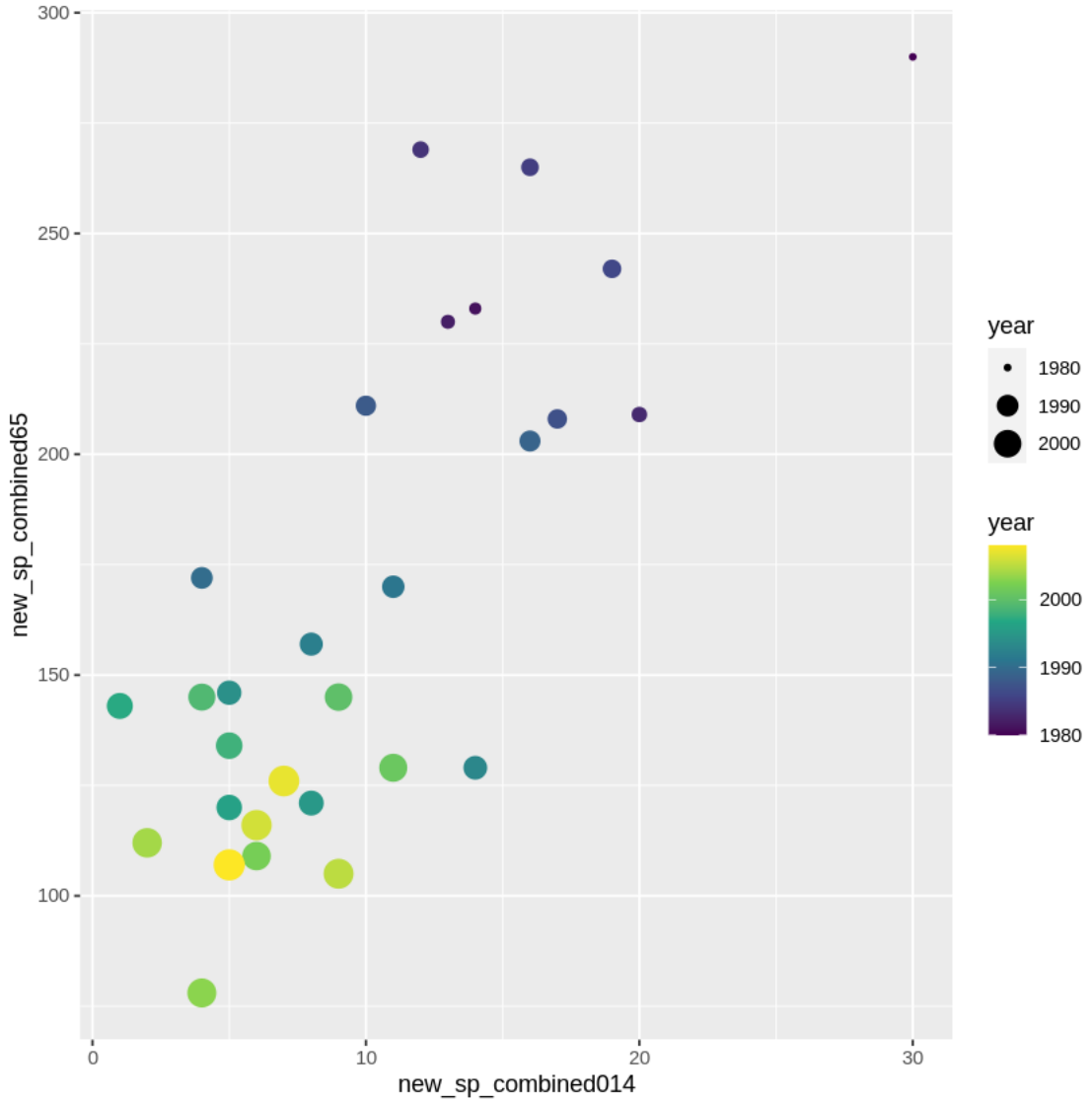
J. Using vector math, create a new variable by combining the numbers from **new_sp_m014** and **new_sp_f014**. Save the resulting vector as a new variable in the **tbCan** df called **new_sp_combined014**. This new variable represents the number of positive pulmonary smear tests for male AND female children between the ages of 0 and 14 years of age. Do the same for SP **tests among citizens 65 years of age and older** and save the resulting vector in the tbCan variable called **new_sp_combined65**.

[64]:
```
tbcan %>%
mutate(new_sp_combined014 = new_sp_m014 + new_sp_f014,
       new_sp_combined65 = new_sp_m65 + new_sp_f65) -> tbcan
```

K. Finally, create a **scatter plot**, showing **new_sp_combined014** on the x axis, **new_sp_combined65** on the y axis, and having the **color and size** of the point represent **year**.

[67]:
```
tbcan %>%
ggplot()+
geom_point(aes(new_sp_combined014,new_sp_combined65, color = year, size =␣
 ↪year))+
scale_color_viridis_c()
```

```
Warning message:
"Removed 23 rows containing missing values (geom_point)."
```



L. Interpret this visualization – what insight does it provide?

[68]:
```
#there are fewer positive tests in the combined014 group than that of the␣
↪combined65 - you can tell purely by the scales of the axes
```