Introduction to R and data analysis

GHP 351 Epidemiology

4/3/2019

Part I - Read datafiles into R

First, we will learn the first steps of any data analysis: reading in the data set and summarizing it.

Topics and Concepts Covered

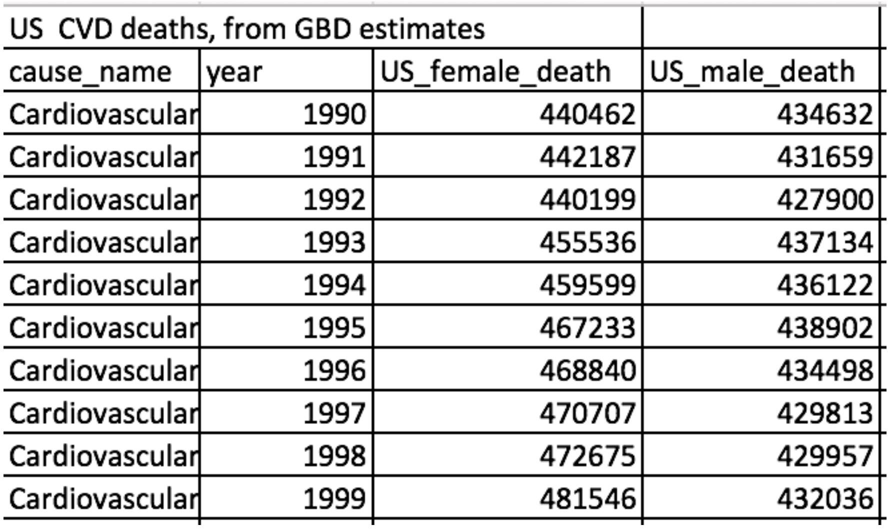
- Loading in data
- Basic operations on vectors
- The structure and format of R

R Commands Covered

- Reading data using `read.csv(..., header = TRUE)` and `read.table(..., header = TRUE)`
- Summarizing the data using `head`, `summary`, and `names`
- Accessing help files using `?` and `??`
- Creating a vector using `c` and accessing its elements using `[` and `]`
- Using `$` to extract columns from a data frame
- Using brackets with two arguments, `[row, column]` to recover elements of a data frame
- Basic operations on vectors: `length`, `mean`, `median`, `min`, `max`, `range`, `sum`, `prod`, `log`

Before beginning this handout. You can make a new folder and set your working directory!

We are going to begin with the cause-of-deaths data for United States from Global Burden of Diseases IHME and Evaluation (n.d.). The data of deaths from cadiovarscular diseases in US for all ages is reprinted below.



US CVD deaths, from GBD estimates

cause_name	year	US_female_death	US_male_death
Cardiovascular	1990	440462	434632
Cardiovascular	1991	442187	431659
Cardiovascular	1992	440199	427900
Cardiovascular	1993	455536	437134
Cardiovascular	1994	459599	436122
Cardiovascular	1995	467233	438902
Cardiovascular	1996	468840	434498
Cardiovascular	1997	470707	429813
Cardiovascular	1998	472675	429957
Cardiovascular	1999	481546	432036

Figure 1: CVD deaths in US

Read in Data

```
cvd <- read.csv("data/cvdDeathsUS.csv", skip = 1, header = TRUE)
# Try to find out what "skip" or "header" is used for
?read.csv
# Take a look at the data
head(cvd)
```

```
                cause_name year US_female_death US_male_death
1 Cardiovascular diseases 1990          440462        434632
2 Cardiovascular diseases 1991          442187        431659
3 Cardiovascular diseases 1992          440199        427900
4 Cardiovascular diseases 1993          455536        437134
5 Cardiovascular diseases 1994          459599        436122
6 Cardiovascular diseases 1995          467233        438902
```

```
# We use read.table for .txt file
# Summarize data using summary()
summary(cvd)
```

```
                    cause_name      year      US_female_death
 Cardiovascular diseases:28   Min.   :1990   Min.   :412391  
                              1st Qu.:1997   1st Qu.:428168  
                              Median :2004   Median :444459  
                              Mean   :2004   Mean   :447194  
                              3rd Qu.:2010   3rd Qu.:468358  
                              Max.   :2017   Max.   :481546  
 US_male_death   
 Min.   :393306  
 1st Qu.:406498  
 Median :427300  
 Mean   :421411  
 3rd Qu.:432652  
 Max.   :455539  
```

```
# names() function will return the vector of column names of the data
names(cvd)
```

```
[1] "cause_name"      "year"            "US_female_death" "US_male_death"
```

Notes

We set the parameter `header` to `TRUE` to let R know that the first row of the file should be used to name each column.

Coding Tip

If you want to access RStudio's help functions, click on the 'Help' tab in the lower right hand box, and type in your question. If you want to access help files from the command line, type in `?command`, e.g. `?sum` to learn about the `sum` function. If you do not remember the exact name of a command, type `??summar` and R will use 'fuzzy matching' to suggest some commands you might be looking for.

Creating Vectors

A vector is simply a collection of numbers or strings. Vectors are constructed in R from using the `c` function, which is used to *combine* objects into a single vector. We are going to create a set of vectors, one for each column in Trende's figure.

```
# Trival example - create a vector of first 10 positive integers
simple_vector <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
simple_vector
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
# or we can use seq() function to do the same
simple_vector_rep <- seq(1:10)
simple_vector_rep
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
# The difference of the two vector should be 10 zeros
difference <- simple_vector - simple_vector_rep
difference
```

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

Attricutes of Data frame

A data frame is used for storing data tables. It is a list of vectors of equal length. Our cvd object is a data frame. `Adata.frame` contains one column for each variable, and one row for eachobservation. Below are some basic operations on data frames.

```
dim(cvd) # dimension of a data frame
```

```
[1] 28  4
```

```
ncol(cvd)  # number of columns
```

```
[1] 4
```

```
nrow(cvd)  # number of rows
```

```
[1] 28
```

We can extract columns of a data frame several different ways. For example, we can use `$` to extract the column directly by name and work with it.

```
# We can use $ to select a column which is a vector in a data frame
cvd$year
```

```
 [1] 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003
[15] 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017
```

```
# We can create new vectors in the dataframe use $ too
cvd$US_both_death <- cvd$US_female_death + cvd$US_male_death
head(cvd)
```

```
               cause_name year US_female_death US_male_death US_both_death
1 Cardiovascular diseases 1990          440462        434632        875094
2 Cardiovascular diseases 1991          442187        431659        873846
3 Cardiovascular diseases 1992          440199        427900        868099
4 Cardiovascular diseases 1993          455536        437134        892670
```

```
5 Cardiovascular diseases 1994          459599        436122        895721
6 Cardiovascular diseases 1995          467233        438902        906135
```

```
# Now we have a new column in the dataframe to indicate the sum
# of CVD deaths from both sexes row by row.
```

Select elements from a vector

We can recall any element of a vector by using brackets.

```
# To see the 10th element of the year vector
cvd$year[10]
```

```
[1] 1999
```

```
# To get the difference of deaths between sexes in year 1990
cvd$US_female_death[1] - cvd$US_male_death[1]
```

```
[1] 5830
```

Select elements from a dataframe

We can select an element from from a dataframe using different ways.

```
# To select the second row of the data set
cvd[2,]
```

```
               cause_name year US_female_death US_male_death US_both_death
2 Cardiovascular diseases 1991          442187        431659        873846
```

```
# To select the second column of the data
# all three following ways returns you the same column
cvd[,2]
```

```
 [1] 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003
[15] 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017
```

```
cvd[,"year"]
```

```
 [1] 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003
[15] 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017
```

```
cvd$year
```

```
 [1] 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003
[15] 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017
```

```
# We want to select an element in a dataframe
cvd[2,2]  # the year in the second row
```

```
[1] 1991
```

```
cvd[2,3]  # US female deaths in year 1991
```

```
[1] 442187
```

```
cvd[6,"US_male_death"] # US male deaths in year 1995
```

```
[1] 438902
```

Question 1A

Try to select an entire column of US_both_death. Try to select the 15th row. Try to select US_female_death in year 2015.

Common functions

R allows us to perform any number of operations on a vector or a dataframe:

```r
length(cvd$year) # Number of elements in a vector
```

```
[1] 28
```

```r
mean(cvd$US_both_death) # Average of the elements
```

```
[1] 868604.5
```

```r
median(cvd$US_both_death) # Middle element, when sorted
```

```
[1] 874470
```

```r
min(cvd$US_both_death) # Smallest element
```

```
[1] 805697
```

```r
max(cvd$US_both_death) # Largest element
```

```
[1] 913582
```

```r
range(cvd$US_both_death) #Range of a vector
```

```
[1] 805697 913582
```

```r
sum(cvd$US_both_death) # Adds all of the numbers in a vector
```

```
[1] 24320926
```

```r
prod(cvd$US_both_death) # Multiplies all of the numbers in a vector
```

```
[1] 1.894009e+166
```

```r
log(cvd$US_both_death) # natural log transformation of all the elements in a vector
```

```
 [1] 13.68209 13.68066 13.67406 13.70197 13.70538 13.71694 13.71385
 [8] 13.71073 13.71307 13.72513 13.72273 13.71867 13.71213 13.69823
[15] 13.66296 13.66174 13.64405 13.62579 13.62189 13.61056 13.59946
[22] 13.61378 13.61841 13.62946 13.64158 13.66150 13.68833 13.71267
```

Question 1B

Try to find the mean, median, sum, minimun and maximum value of US_female_death and US_male_death respectively.

Changing Names to a Vector

Next, we can give or change names to a vector:

```
# we can also *take a copy* of the column and work with this new vector
cvdDeathsFemale <- cvd$US_female_death
names(cvdDeathsFemale) # No names yet
```

```
NULL
```

```
names(cvdDeathsFemale) <- paste("female", cvd$year, sep = "")
cvdDeathsFemale
```

```
female1990 female1991 female1992 female1993 female1994 female1995
    440462     442187     440199     455536     459599     467233
female1996 female1997 female1998 female1999 female2000 female2001
    468840     470707     472675     481546     481252     480917
female2002 female2003 female2004 female2005 female2006 female2007
    475435     468198     451019     448122     438949     429023
female2008 female2009 female2010 female2011 female2012 female2013
    425602     419646     412391     418039     417611     419241
female2014 female2015 female2016 female2017
    422962     429440     437860     446731
```

```
cvdDeathsFemale["female2016"]
```

```
female2016
    437860
```

Question 1C

Try to figure out what ‘paste()‘ function does using R documentation or Stack Overflow or Google.

Part II - Subset data

In this second part, we will learn how to subset data, a crucial step in adjusting for confounders, as well as how to calculate statistics for subsets of the data.

Topics and Concepts Covered

- Subsetting vectors
- Subsetting data frames
- Creating tables
- Conditional statements
- Calculating sum/mean by subgroups

R Commands Covered

- Subsetting a vector using `[` and `]`
- Boolean operators for subsetting: `>`, `>=`, `<`, `<=`, and `==`
- Creating multiple conditions through using `&` and `|`
- Learning the class of a vector (`factor`, `numeric`, `character`) with `class`
- Converting numeric vectors to factors with `as.factor`
- Setting the levels of a factor with `levels`

Boolean (Logical) Operators Commonly Used for Subsetting

- `>`, `>=`. Greater than, greater than or equal to
- `<`, `<=`. Less than, less than or equal to
- `==`. Exactly equal to
- `&`. And
- `|`. Or

Examples

Return all elements of `y` for which `x` is less than (less than or equal to) 2

```
y[ x < 2]
y[x <= 2]
```

Return all elements of `y` for which `x` is exactly equal to 2

```
y[ x == 2]
```

Return all elements of `y` for which `x` is exactly equal to `z`

```
y[ x == z]
```

Return all elements of `y` for which `x` is greater than 2 *and* `x` is less than 5

```
y[(x > 2) & (x < 5)]
```

Return all elements of `y` for which `x` is less than or equal to 2 *or* `x` is greater than 5

```
y[(x <= 2) | (x > 5)]
```

Subsetting Data

Subsetting data is going to be a crucial component of this course. We are going to explore how the relationship between two variables changes as we move from one subset of the data to another, and use this information to draw inferences.

Numeric, Factor, and Character Variables

In order to learn how to select subsets of data, we are going to begin with subsetting a single vector. When we *subset* a vector, we are looking only at a portion of the vector that satisfies certain conditions.

First, we are going to load in the data and make sure that it loaded properly.

```
deaths <- read.csv("data/deaths.csv", header = TRUE)
head(deaths)
```

```
  location_id location_name sex_id sex_name age_id    age_name cause_id
1           6         China      1     Male      1     Under 5      298
2           6         China      2   Female      1     Under 5      298
3           6         China      1     Male     23  5-14 years      298
4           6         China      2   Female     23  5-14 years      298
5           6         China      1     Male     24 15-49 years      298
6           6         China      2   Female     24 15-49 years      298
  cause_name year deaths_number upper_deaths_number lower_deaths_number
1   HIV/AIDS 2010           611                 691                 546
2   HIV/AIDS 2010           471                 536                 414
3   HIV/AIDS 2010           118                 138                  99
4   HIV/AIDS 2010            95                 112                  79
5   HIV/AIDS 2010          9298                9597                9041
6   HIV/AIDS 2010          3400                3498                3315
```

Numeric Variables

First we are interested in how many countries are included in the dataset. let's look at the variable `location_id`.

```
class(deaths$location_id)
```

```
[1] "integer"
```

```
mean(deaths$location_id)
```

```
[1] 106
```

You will see that the `location_id` variable has a class of `integer`. R interprets variables with the class `numeric` and `integer` as quantitative variables. Therefore, we can perform basic arithmetic operations, such as taking the mean or median on variables of this type, but the mean of this variable would not make any sense as we can assign location ID to countries arbitrarily.

In this case, we first want to create a variable that is a nominal variable, which R calls a `factor`. To do this, first check the class of the location_id variable.

Factors

R maintains a special class of variable for nominal variables. These variables, called `factors`, are simply interpreted as a set of levels. So, even if R returns levels of 1, 2, 3, etc., these are interpreted as categories, and not numbers. Let's look at an example, where we use the command `as.factor` to create a version of race that is a factor:

```r
deaths$location_id2 <- as.factor(deaths$location_id)
head(deaths)
```

```
  location_id location_name sex_id sex_name age_id    age_name cause_id
1           6         China      1     Male      1     Under 5      298
2           6         China      2   Female      1     Under 5      298
3           6         China      1     Male     23  5-14 years      298
4           6         China      2   Female     23  5-14 years      298
5           6         China      1     Male     24 15-49 years      298
6           6         China      2   Female     24 15-49 years      298
  cause_name year deaths_number upper_deaths_number lower_deaths_number
1   HIV/AIDS 2010           611                 691                 546
2   HIV/AIDS 2010           471                 536                 414
3   HIV/AIDS 2010           118                 138                  99
4   HIV/AIDS 2010            95                 112                  79
5   HIV/AIDS 2010          9298                9597                9041
6   HIV/AIDS 2010          3400                3498                3315
  location_id2
1            6
2            6
3            6
4            6
5            6
6            6
```

Notice how `location_id2` was appended to the last column of the data frame `deaths`. To check the class of `location_id2`, we can do the following:

```r
class(deaths$location_id2)
```

```
[1] "factor"
```

```r
levels(deaths$location_id2)
```

```
[1] "6"   "67"  "80"  "81"  "102" "135" "163" "214"
```

```r
# mean(deaths$location_id2) # Run this line -- it returns an NA
```

Notice that the mean of a factor is nonsensical; R returns `NA`. The levels of the variable `location_id2` are ids of each level. We can make these levels more informative using information from Country names (`location_name`).

```r
class(deaths$location_name) # it's already a factor
```

```
[1] "factor"
```

```r
levels(deaths$location_name) # shows the country names
```

```
[1] "Brazil"        "China"         "France"        "Germany"      
[5] "India"         "Japan"         "Nigeria"       "United States"
```

```
table(deaths$location_name)   # table returns us the frequency of data entries in each country
```

```
       Brazil         China        France       Germany         India 
          160           160           160           160           160 
        Japan       Nigeria United States 
          160           160           160 
```

```
# We can make location_id2 levels more informative using country codes (ISO3).
levels(deaths$location_id2) <- c("CHN", "JPN", "FRA", "DEU", "USA", "BRA", "IND", "NGA")
head(deaths)
```

```
  location_id location_name sex_id sex_name age_id    age_name cause_id
1           6         China      1     Male      1     Under 5      298
2           6         China      2   Female      1     Under 5      298
3           6         China      1     Male     23   5-14 years      298
4           6         China      2   Female     23   5-14 years      298
5           6         China      1     Male     24 15-49 years      298
6           6         China      2   Female     24 15-49 years      298
  cause_name year deaths_number upper_deaths_number lower_deaths_number
1   HIV/AIDS 2010           611                 691                 546
2   HIV/AIDS 2010           471                 536                 414
3   HIV/AIDS 2010           118                 138                  99
4   HIV/AIDS 2010            95                 112                  79
5   HIV/AIDS 2010          9298                9597                9041
6   HIV/AIDS 2010          3400                3498                3315
  location_id2
1          CHN
2          CHN
3          CHN
4          CHN
5          CHN
6          CHN
```

The variable `location_id2` now displays in a more informative manner. We are using these abbreviations so that the labels fit on the figures and tables we are going to produce.

Question 2A

Try to find out how many age categories in the data set? How many causes-of-death in the dataset? How many years are included in the dataset?

Character

Occasionally, you may see a variable that has a class of `character`. In this case, R is interpreting the variable as a string of text. Variables in this form are most commonly used to label axes or figures, not for analysis. To turn the variable `location_id2` into a character vector, use

```
char_location <- as.character(deaths$location_id2)
char_location[1:5] # Look at the first 5 elements
```

```
[1] "CHN" "CHN" "CHN" "CHN" "CHN"
```

The quotation marks indicate that the vector is being interpreted as a character.

Coding Tip

Occasionally, when R reads in data, it will interpret a variable as having class character or factor when you want it to be numeric. We can use `as.numeric` to coerce a variable to be interpreted as numeric.

If you have a variable that is numeric that you want to be a factor, you can use `as.factor` to coerce the variable into a factor.

Subsetting a Vector

In this section, we are going to look at subsets of a vector. In order to produce a subset of a vector, R uses the following syntax:

```
variable[ condition ]
```

where `variable` is the name of some variable, and `condition` is an expression saying what observations you want to look at.

Condition (==)

Let's first find out the total deaths number in the dataset

```r
sum(deaths$deaths_number)
```

```
[1] 70873610
```

Then let's find out the total female deaths and total male deaths in the dataset

```r
# Female deaths
sum(deaths$deaths_number[deaths$sex_name == "Female"])
```

```
[1] 32829685
```

```r
# Male deaths
sum(deaths$deaths_number[deaths$sex_name == "Male"])
```

```
[1] 38043925
```

Similary let's find out the total deaths caused by HIV/AIDS and CVD respectively. Which one causes more deaths?

```r
# HIV/AIDS
sum(deaths$deaths_number[deaths$cause_name == "HIV/AIDS"])
```

```
[1] 2632404
```

```r
# CVD
sum(deaths$deaths_number[deaths$cause_name == "Cardiovascular diseases"])
```

```
[1] 68241206
```

Question 2B

Try to find out the total death numbers in each country in this dataset.

Condition (>,<, >=, <=)

First let's check variable `age_id`

```
class(deaths$age_id) # it's numeric although age_name is categorical
```

```
[1] "integer"
```

```
class(deaths$age_name)
```

```
[1] "factor"
```

```
levels(deaths$age_name) # age_id increases for an older category
```

```
[1] "15-49 years" "5-14 years"  "50-69 years" "70+ years"   "Under 5"
```

Let's find out the total deaths in this dataset Under age 5 - age code is 1 here - less than 20

```
# The following three lines will produce the same result
sum(deaths$deaths_number[deaths$age_id == 1])
```

```
[1] 288835
```

```
sum(deaths$deaths_number[deaths$age_id <  2])
```

```
[1] 288835
```

```
sum(deaths$deaths_number[deaths$age_id <= 1])
```

```
[1] 288835
```

Let's find out the total deaths in this dataset above age 50 [50-69(`age_id` == 25) & age 70+(`age_id` == 26)]

```
# The following three lines will produce the same result
sum(deaths$deaths_number[deaths$age_id >= 25])
```

```
[1] 63628915
```

```
sum(deaths$deaths_number[deaths$age_id >  24])
```

```
[1] 63628915
```

```
sum(deaths$deaths_number[deaths$age_id == 25 | deaths$age_id == 26])
```

```
[1] 63628915
```

Condition (|, &)

| stands for `or`, and & stands for and. We can use combine several logical statements. For instance, let's find out the total female deaths in this dataset under age 5 in France.

```
sum(deaths$deaths_number[deaths$sex_name == "Female"
                         & deaths$location_name == "France" & deaths$age_id <= 1])
```

```
[1] 214
```

let's find out the total deaths in this dataset in China or India.

```
sum(deaths$deaths_number[deaths$location_name == "China"
                         | deaths$location_name == "China"])
```

```
[1] 32254249
```

```
# Let's add another condition of deaths in 2015
sum(deaths$deaths_number[(deaths$location_name == "China"
                          | deaths$location_name == "China") & deaths$year == 2015])
```

```
[1] 4250781
```

Question 2C

Try to find out the total deaths caused by HIV/AIDS in Nigeria in 2016 or 2017.

Subsetting a Data Frame

Now that we know how to subset a vector, we may want to subset a data frame. This allows us to subset all of the columns in a data frame simultaneously. The syntax for subsetting a data frame is

```
data[rows, columns]
```

where the first argument in the brackets tells you what rows to consider and the second tells what columns. As we learned in Part I, we wanted the third row of `deaths`, we would use

```
deaths[3, ]
```

```
  location_id location_name sex_id sex_name age_id   age_name cause_id
3           6         China      1     Male     23 5-14 years      298
  cause_name year deaths_number upper_deaths_number lower_deaths_number
3   HIV/AIDS 2010           118                 138                  99
  location_id2
3          CHN
```

while if we wanted the value of the 1003rd row and fifth column, we would use

```
deaths[1003, 5]
```

```
[1] 24
```

Often, we will want to subset an entire data frame. As an example, let's say we wanted to consider the subset of `deaths` for only females. In this case, we would subset the data as

```
deathsFemale <- deaths[deaths$sex_name == "Female", ]
head(deathsFemale)
```

```
   location_id location_name sex_id sex_name age_id    age_name cause_id
2            6         China      2   Female      1     Under 5      298
4            6         China      2   Female     23  5-14 years      298
6            6         China      2   Female     24 15-49 years      298
8            6         China      2   Female     25 50-69 years      298
10           6         China      2   Female     26   70+ years      298
12          81       Germany      2   Female      1     Under 5      491
                cause_name year deaths_number upper_deaths_number
2                 HIV/AIDS 2010           471                 536
4                 HIV/AIDS 2010            95                 112
6                 HIV/AIDS 2010          3400                3498
8                 HIV/AIDS 2010          1234                1295
10                HIV/AIDS 2010           253                 270
12 Cardiovascular diseases 2010            15                  19
   lower_deaths_number location_id2
2                  414          CHN
```

```
4                   79          CHN
6                 3315          CHN
8                 1167          CHN
10                 239          CHN
12                  13          DEU
```

```r
# We can compare the dimension of the data before and after the subsetting
dim(deaths)
```

```
[1] 1280   13
```

```r
dim(deathsFemale)
```

```
[1] 640  13
```

which is telling R to take all of the rows of `deaths` for which `sex_name == "Female"` (the first argument) and then return all columns (the second argument).

Here's the first four columns:

```r
deathsFemaleFourCol <- deaths[deaths$sex_name == "Female", 1:4]
head(deathsFemaleFourCol)
```

```
   location_id location_name sex_id sex_name
2            6         China      2   Female
4            6         China      2   Female
6            6         China      2   Female
8            6         China      2   Female
10           6         China      2   Female
12          81       Germany      2   Female
```

We can use multiple conditions to subset dataframes too. For instance, we want to create a subset of Germany's deaths caused from Cardiovascular diseases in 2010.

```r
deathsDEUcvd2000 <- deaths[deaths$location_name == "Germany"
                           & deaths$cause_name == "HIV/AIDS" & deaths$year == 2010, ]
head(deathsDEUcvd2000)
```

```
   location_id location_name sex_id sex_name age_id    age_name cause_id
61          81       Germany      1     Male      1     Under 5      298
62          81       Germany      2   Female      1     Under 5      298
63          81       Germany      1     Male     23  5-14 years      298
64          81       Germany      2   Female     23  5-14 years      298
65          81       Germany      1     Male     24 15-49 years      298
66          81       Germany      2   Female     24 15-49 years      298
   cause_name year deaths_number upper_deaths_number lower_deaths_number
61   HIV/AIDS 2010             3                   3                   2
62   HIV/AIDS 2010             4                   5                   4
63   HIV/AIDS 2010             3                   3                   2
64   HIV/AIDS 2010             2                   2                   1
65   HIV/AIDS 2010           240                 253                 227
66   HIV/AIDS 2010            68                  73                  64
   location_id2
61          DEU
62          DEU
63          DEU
64          DEU
65          DEU
66          DEU
```

Question 2D

Try to create a subset of deaths caused from Cardiovascular diseases in Japan and United States from year 2011 to 2015.

Two-way table and summary by subgroups

If we have two categorical variables, we can use two-way table to find out frequencies in each subgroup.

```
table(deaths$location_id2, deaths$year)
```

```
      2010 2011 2012 2013 2014 2015 2016 2017
  CHN   20   20   20   20   20   20   20   20
  JPN   20   20   20   20   20   20   20   20
  FRA   20   20   20   20   20   20   20   20
  DEU   20   20   20   20   20   20   20   20
  USA   20   20   20   20   20   20   20   20
  BRA   20   20   20   20   20   20   20   20
  IND   20   20   20   20   20   20   20   20
  NGA   20   20   20   20   20   20   20   20
```

We can also use `aggregate()` function to find summary statistics for subgroups. For instance, we can find the sum or the mean of deaths in each country or age group with one line codes.

```
deathSumByCountry <- aggregate(deaths$deaths_number,
                               by=list(Category= deaths$location_id2), FUN=sum)
deathSumByCountry
```

```
  Category        x
1      CHN 32254249
2      JPN  2751274
3      FRA  1203085
4      DEU  2716321
5      USA  6816859
6      BRA  2947965
7      IND 19766303
8      NGA  2417554
```

```
#Deaths caused by HIV/AIDS by country in eight years
HIVdeathSumByCountry <- aggregate(deaths$deaths_number[deaths$cause_name == "HIV/AIDS"],
                 by=list(Category= deaths$location_id2[deaths$cause_name == "HIV/AIDS"]), FUN=sum)
HIVdeathSumByCountry
```

```
  Category       x
1      CHN  198216
2      JPN    1813
3      FRA    4067
4      DEU    4188
5      USA   62050
6      BRA  122615
7      IND  711734
8      NGA 1527721
```

```
# Mean HIV/AIDS deaths in each age group (average by sex and country)
HIVdeathMeanByAgegroup <- aggregate(deaths$deaths_number[deaths$cause_name == "HIV/AIDS"],
                    by=list(Category= deaths$age_name[deaths$cause_name == "HIV/AIDS"]), FUN=mean)
HIVdeathMeanByAgegroup
```

```
     Category          x
1 15-49 years 14942.5625
2  5-14 years   428.7656
3 50-69 years  3322.4766
4   70+ years   344.5156
5     Under 5  1527.3359
```

Question 2E

Try to use aggregate to find out the total female deaths in this dataset from HIV/AIDS by year.

Question 2F

There are many ways to get aggregate summary statistics for subgroups. Use Google and Stack Overflow to find another way to achieve the same goal in 2E.

Part III - Plot

In the last part of this introduction, we will learn how to plot points and lines, as well as how to create density plots and box plots. Data visualization is a great way to convey information about complex data.

Topics and Concepts Covered

- Creating box-and-whisker plots
- Placing multiple plots on one figure
- Producing density plots with multiple densities in the same figure

R Commands Covered

- Calculating a statistic by groups of data using `tapply`
- Creating variables through using the conditional statement `ifelse`
- Creating box-and-whisker plots through `boxplot`
- Placing multiple plots in one figure by setting `mfrow` with the `par` function
- Using `plot` and `lines` to create and add lines to a figure
- Using `density` to make density plots
- Using `legend` to add a legend to a plot

Summary of Options Used in Figures

Here are some parameters with example values you can use with `plot`, `lines`, `points`:

- `main = "Distribution of Wealth"`. Set the main figure title.
- `xlab = "Time"`. Set the x-axis label.
- `ylab = "Density"`. Set the y-axis label.
- `xlim = c(-1, 2.4)`. Constrain the x-axis to start at -1 and end at 2.4.
- `ylim = c(0, 1)`. Constrain the y-axis to start at 0 and end at 1.
- `lty = 2`. Change the line type. 1 is solid, 2 is dashed, and 3–5 are different types of dashed lines.
- `pch = 19`. Set the plotting character of points. 1 is an unfilled circle. See the help page for `points` for the complete list.
- `col = "red"`. Set the line or point color. You can provide more than one!

Legends

`legend` adds a legend to your plot. It has the following arguments:

- The first argument is the legend's location. Choose one of 'topleft', 'bottomleft', 'topright', or 'bottom right'.
- `legend` is a vector of character strings, indicating what the legend should contain.
- `col`. A vector of colors, corresponding to the elements of `legend`.
- `lty`. A vector of line types, corresponding with the elements of `legend`.
- `bg = "grey"`. Turns the background of the legend from the default background color, usually white, to grey.

Calculating Statistics for Subsets of the Data

First, we used `aggregate()` function to calculate statistics for subgroups in Part II. Now we are going to use the command `tapply`. This function takes three arguments:

- `X`. A variable to which we want to apply a function
- `INDEX`. A variable defining the groups within which we want to apply the function
- `FUN`. The function we want to apply

For example, if we wanted to calculate the total deaths in this dataset by country, we could

```
tapply(deaths$deaths_number, INDEX = deaths$location_name, FUN = sum)
```

```
       Brazil         China        France       Germany         India 
      2947965      32254249       1203085       2716321      19766303 
        Japan       Nigeria United States 
      2751274       2417554       6816859 
```

With even a modest number of groups, `tapply` can prove quite useful. If we wanted the sum of detahs for each country, caused by HIV/AIDS and CVD respectively, we would:

```
#Total deahts caused by HIV/AIDS
deathsHIV <- tapply(deaths$deaths_number[deaths$cause_name == "HIV/AIDS"],
       INDEX = deaths$location_name[deaths$cause_name == "HIV/AIDS"], FUN = sum)
deathsHIV
```

```
       Brazil         China        France       Germany         India 
       122615        198216          4067          4188        711734 
        Japan       Nigeria United States 
         1813       1527721         62050 
```

```
#CVD - 491 is cause_id for CVD
deathsCVD <- tapply(deaths$deaths_number[deaths$cause_id == 491],
       INDEX = deaths$location_name[deaths$cause_id == 491], FUN = sum)
deathsCVD
```

```
       Brazil         China        France       Germany         India 
      2825350      32056033       1199018       2712133      19054569 
        Japan       Nigeria United States 
      2749461        889833       6754809 
```

```
deathsHIV - deathsCVD
```

```
       Brazil         China        France       Germany         India 
     -2702735     -31857817      -1194951      -2707945     -18342835 
        Japan       Nigeria United States 
     -2747648        637888      -6692759 
```

From the different of the deaths from two causes, most countries in our data havemore deaths caused by CVD than HIV, except Nigeria having more HIV deaths than CVD deaths.

We can put any function into `tapply` that we like: `sd`, `median`, and so on.
`tapply` can also be used to calculate the number of observations in each category, by setting `FUN = length`. For example

```
tapply(deaths$upper_deaths_number,
       INDEX = deaths$location_id2,
       FUN = length)
```

```
CHN JPN FRA DEU USA BRA IND NGA 
```

```
160 160 160 160 160 160 160 160
```

```
# Try to find how many unique years in the dataset
length(unique(deaths$year))
```

```
[1] 8
```

```
range(deaths$year)
```

```
[1] 2010 2017
```

Every country has 160 entries (`8 years * 5 age groups * 2 sexes * 2 causes of deaths`).

Question 3A

Try to use tapply() to find the standard deviation of CVD deaths across countries in 8 years by sex

Conditional Statements

We may also want to create conditional statements. We do this through the command `ifelse`. The command takes three arguments:

- `test`. A logical expression (one that is either true or false) e.g. `x < 2` or `x == "Japan"`.
- `yes`. What to return if the `test` is TRUE
- `no`. What to return if the ‘test is FALSE

For example, let’s say we wanted to create a variable that took on a value of 1 for observations in most recent half duration (year 2014-2017), and a 0 for observations in earlier half duration(year 2010-2013) . We could do so using

```
deaths$recent <- ifelse(deaths$year > median(deaths$year), 1, 0)
```

We have just created a variable, `recent`, which takes on a value of 1 when `year` is above its median, and 0 when `year` is below its median. If we look at this new variable

```
table(deaths$recent)
```

```

  0   1 
640 640 
```

we can see that we do have half 1’s and half 0’s.

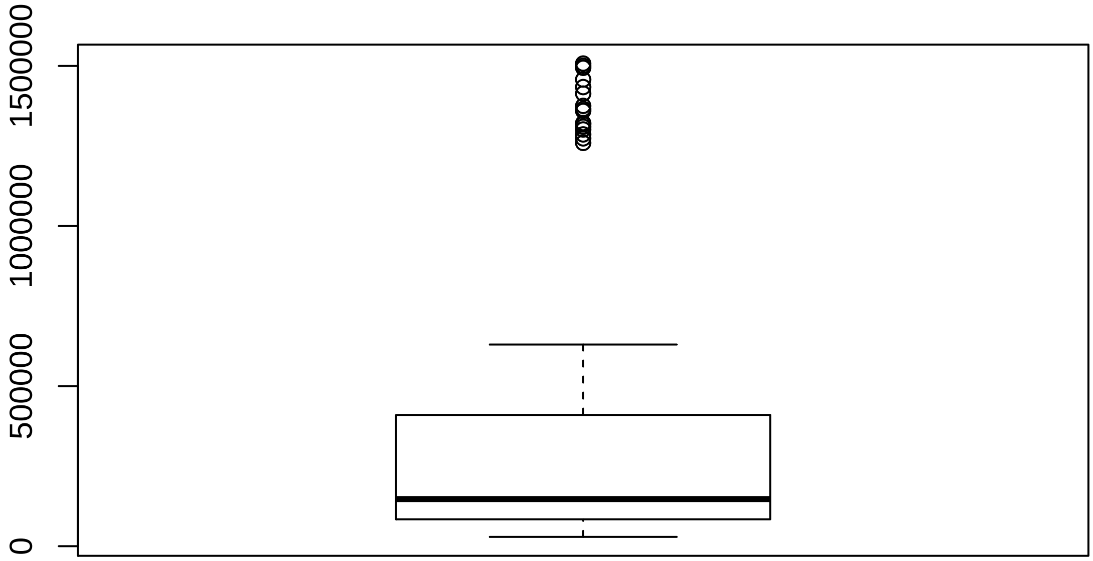
Question 3B

Try to use ifelse() to create a binary variable - 1 is for *lower_deaths_number* more than ‘100,000‘; 0 otherwise. The *lower_deaths_number* is the lower estimates of deaths counts for a specific age group, a cause-of-deaths and a sex in a specific year.

Creating Box-and-Whisker Plots

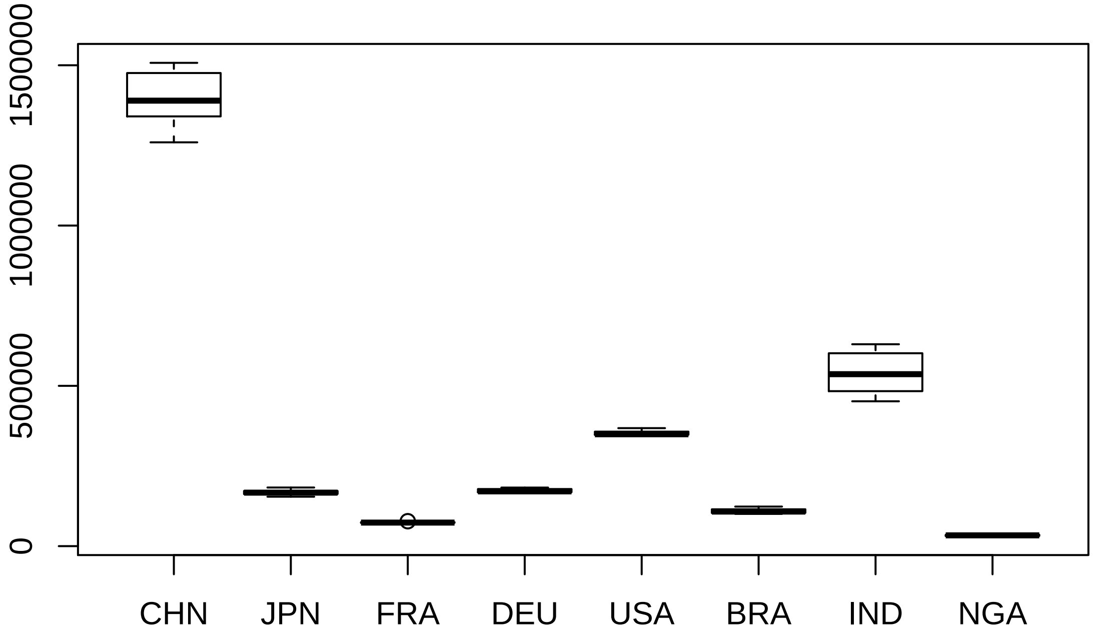
Let’s look at CVD deaths above age 70 across countries. First we need to subset the data. R makes creating box-and-whisker plots straightforward. The command is `boxplot`, and if we place a variable in the function, it returns a boxplot, as

```
# Subset data
CVD70 <- deaths[deaths$cause_id == 491 & deaths$age_id == 26, ]
#boxplot
boxplot(CVD70$deaths_number)
```

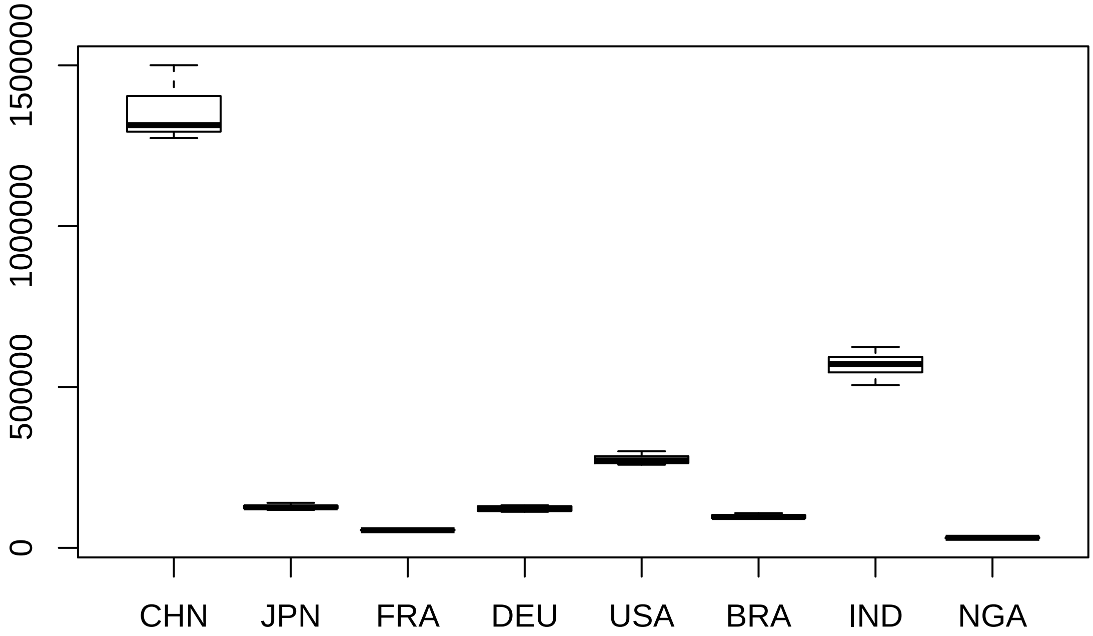


The function `boxplot` can be used to construct separate boxes for the categories of a different variable. For example, let's say we want to look at the box plots of deaths of CVD above age 70 across countries:

```
# Female deaths caused by CVD above age 70
boxplot(CVD70$deaths_number[CVD70$sex_name == "Female"]
        ~ CVD70$location_id2[CVD70$sex_name == "Female"])
```



```
# Male deaths caused by CVD above age 70
boxplot(CVD70$deaths_number[CVD70$sex_name == "Male"]
        ~ CVD70$location_id2[CVD70$sex_name == "Male"])
```

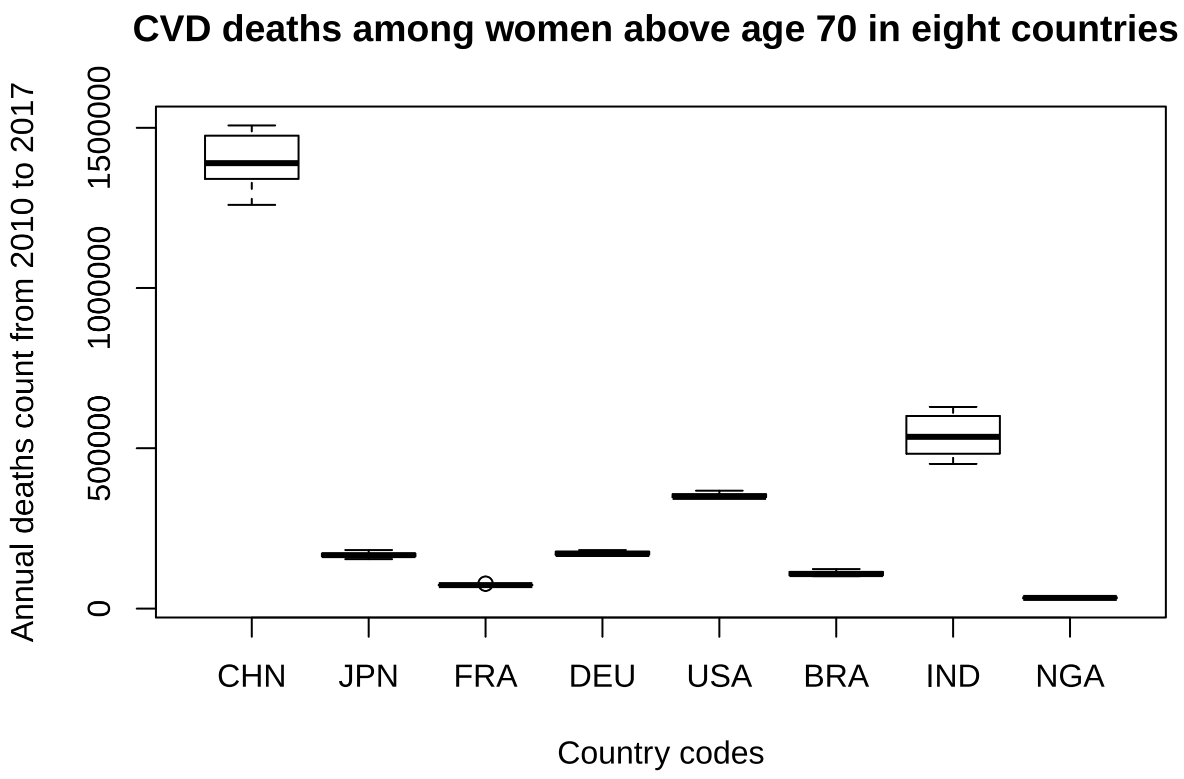


Question 3C

What are the datapoints in our ‘CVD70‘ dataset for the boxplot? Why do the deaths numbers vary a lot across countries? What measurement should we use if we want to make the mortality situation more comparable across countires?

We can give the box plot a title and y-axis label, by setting the `main` and `xlab` and `ylab` parameters to suitable values:

```r
# Female deaths caused by CVD above age 70
boxplot(CVD70$deaths_number[CVD70$sex_name == "Female"]
        ~ CVD70$location_id2[CVD70$sex_name == "Female"],
        main = "CVD deaths among women above age 70 in eight countries ",
        xlab = "Country codes",
        ylab = "Annual deaths count from 2010 to 2017" )
```



Placing Several Plots in One Figure

Finally, we are going to add a command that allows for multiple plots in one figure. To do so, we need adjust R's default plot parameters. We'll use the `par` function to set the value of `mfrow` which controls how plots are arranged in the plotting window. We use `par` to set `mfrow` like this:

par(mfrow = c(*number of rows*, *number of columns*))

and we have to remember to put this line *before* (that is, above) any plots we make.

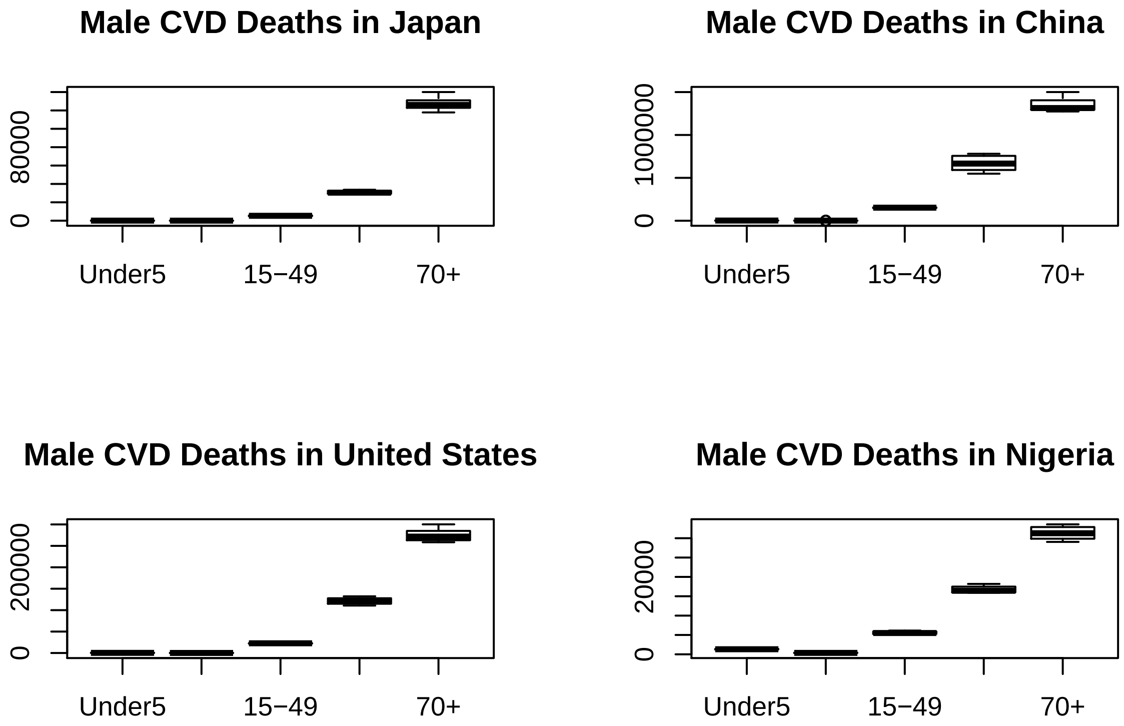
When mfrow is set this way each new plot we make will start in a new part of the figure. To give an example, let's say we wanted to create four box plots in two columns and two rows to get a age distribution of CVD deaths among males in Japan, China, United States, Nigeria.

```
# subset data first
CVDmale <- subset(deaths, sex_name == "Male"
                    & cause_name == "Cardiovascular diseases")
par(mfrow = c(2, 2)) # two rows, two column columns.
boxplot(CVDmale$deaths_number[CVDmale$location_name == "Japan"]
        ~ CVDmale$age_id[CVDmale$location_name == "Japan"],
        main = "Male CVD Deaths in Japan",  xaxt = 'n')
# replace age_id labels with new labels for x-axis.
axis(1, at=1:5, labels= c("Under5", "5-14", "15-49", "50-69","70+"))

boxplot(CVDmale$deaths_number[CVDmale$location_name == "China"]
        ~ CVDmale$age_id[CVDmale$location_name == "China"],
        main = "Male CVD Deaths in China",  xaxt = 'n')
axis(1, at=1:5, labels= c("Under5", "5-14", "15-49", "50-69","70+"))
```

```
boxplot(CVDmale$deaths_number[CVDmale$location_id2 == "USA"]
        ~ CVDmale$age_id[CVDmale$location_id2 == "USA"],
        main = "Male CVD Deaths in United States",  xaxt = 'n')
axis(1, at=1:5, labels= c("Under5", "5-14", "15-49", "50-69","70+"))

boxplot(CVDmale$deaths_number[CVDmale$location_id2 == "NGA"]
        ~ CVDmale$age_id[CVDmale$location_id2 == "NGA"],
        main = "Male CVD Deaths in Nigeria",  xaxt = 'n')
axis(1, at=1:5, labels= c("Under5", "5-14", "15-49", "50-69","70+"))
```



As we can see, all four countries share same age patterns of CVD deaths as CVD deaths occur at an older age.

Question 3D

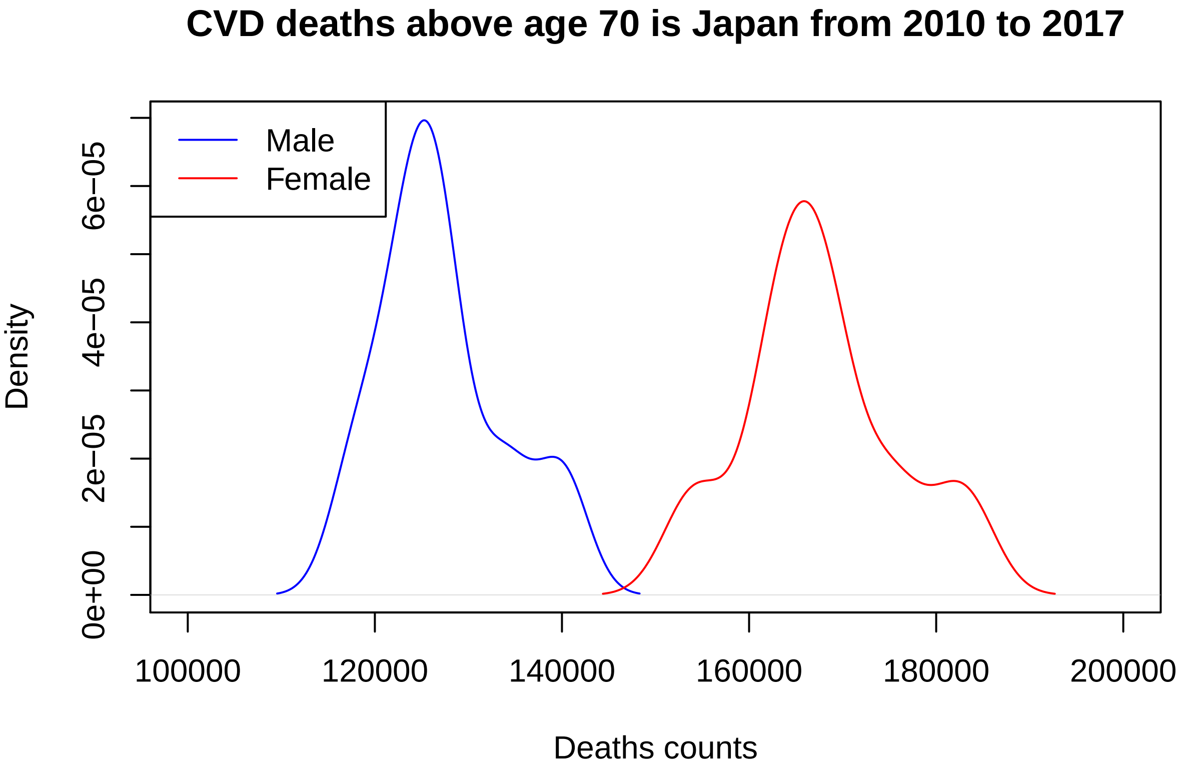
Try to plot another '2*2' table of HIV/AIDS deaths for the same set of countries.

Density plots

We use `plot()` to create the first blue line of male density and use `lines()` to add an additional density line for women in red. `legend()` can specify which line stands for which.

```
# Top left figure -- CVD deaths above age 70 in Japan
plot(density(CVD70$deaths_number[CVD70$location_id2 == "JPN"
                                 & CVD70$sex_name == "Male" ]),
     xlim = c(10e4, 2*10e4), # set limits of x-axis
     main = "CVD deaths above age 70 is Japan from 2010 to 2017",
     xlab = "Deaths counts", col = "blue")
lines(density(CVD70$deaths_number[CVD70$location_id2 == "JPN"
                                  & CVD70$sex_name == "Female" ]), col = "red")
```

```
legend("topleft", c("Male",  "Female"),
       lty = c(1, 1), col = c("blue", "red"))
```



Question 3E

Try to plot the similar figure (CVD deaths above age 70) in other countries. Can you find the same pattern that female deaths are more than male deaths? If yes, how would you explain this sex difference in CVD mortality?

Reference

IHME, Institute for Health Metrics, and Evaluation. n.d. "Global Burden of Disease Study 2017 (Gbd 2017) Results." *Global Burden of Disease Collaborative Network*.