# DATA 106 - Notes 2

*Jillian Morrison*

*9/4/2019*

## Working Directory

Working Directory is the path on your computer where everything is saved or retrieved, unless specified otherwise.

```
getwd() # check current working directory
setwd(path)  #Setting the working directory

#Path should be a folder on your machine for example:
#"C:/Users/jmorrison/OneDrive - The College of Wooster/College of Wooster/Fall2019/Data Analytics/Commo
```

## Import and Export Data

Some files to deal with:

- *.xls file:* Excel file
- *.txt file:* tab separated file
- *.csv file:* comma separated file
- *.RData file:* R data file
- *.sas7bdat file:* SAS file

## read.table

- Reads a file in table format
- creates a data frame
- Use `?read.table` to get more information

Syntax:

```
read.table(file, header = FALSE, na.strings = "NA",
           stringsAsFactors = default.stringsAsFactors(),
           colClasses = NA, skip = 0)
```

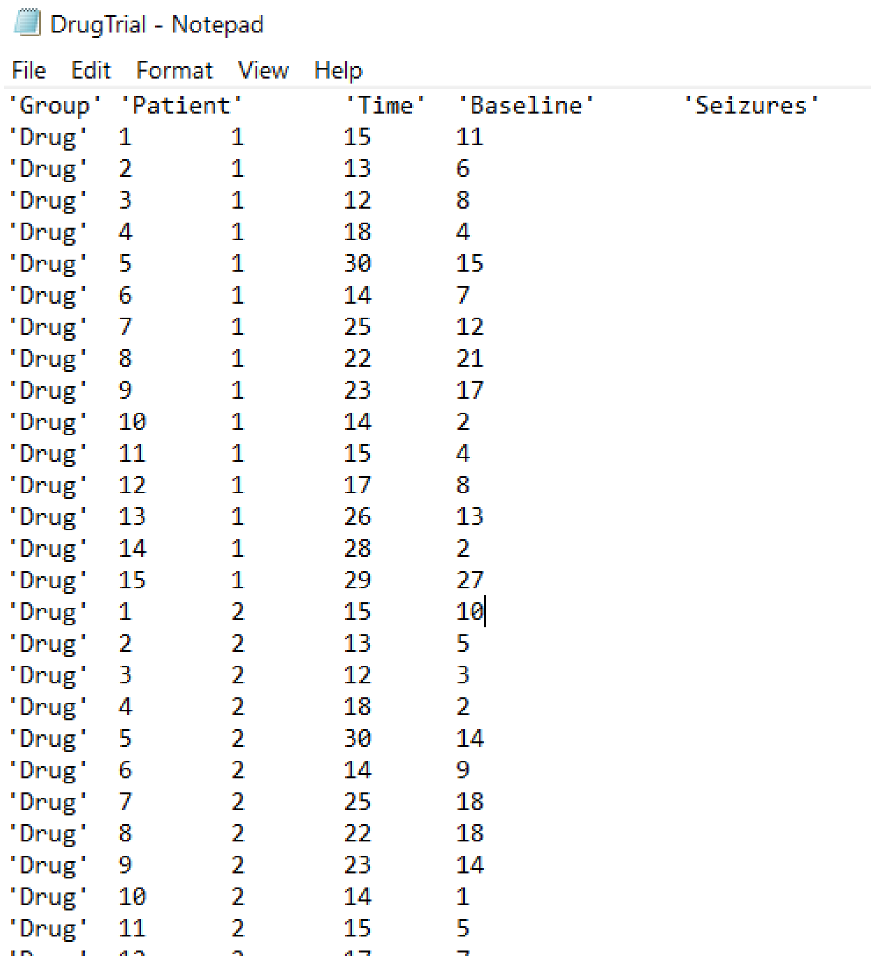
Other options for colClasses: colClasses = c("factor", "character", "integer", "numeric", "Date", "logical")))

For na.strings put replacement for emply cells - e.g. " " for leave empty

## `read.table`

- `header`: a logical value indicating whether the file contains the names of the variables as its first line.
- `na.strings`: a character vector of strings which are to be interpreted as NA values???
- `stringsAsFactors`: logical: should character vectors be converted to factors?
- `colClasses`: character. A vector of classes to be assumed for the columns. "Date" for date can be set via this.
- `skip`: integer: the number of lines of the data file to skip before beginning to read data.

## What a text file looks like



```
'Group' 'Patient'        'Time'  'Baseline'       'Seizures'
'Drug'  1         1       15      11
'Drug'  2         1       13      6
'Drug'  3         1       12      8
'Drug'  4         1       18      4
'Drug'  5         1       30      15
'Drug'  6         1       14      7
'Drug'  7         1       25      12
'Drug'  8         1       22      21
'Drug'  9         1       23      17
'Drug'  10        1       14      2
'Drug'  11        1       15      4
'Drug'  12        1       17      8
'Drug'  13        1       26      13
'Drug'  14        1       28      2
'Drug'  15        1       29      27
'Drug'  1         2       15      10
'Drug'  2         2       13      5
'Drug'  3         2       12      3
'Drug'  4         2       18      2
'Drug'  5         2       30      14
'Drug'  6         2       14      9
'Drug'  7         2       25      18
'Drug'  8         2       22      18
'Drug'  9         2       23      14
'Drug'  10        2       14      1
'Drug'  11        2       15      5
```

## Some properties of txt file

- file name "DrugTrial.txt"
- first row - column names
- observations are numeric, except Group (i.e. no strings or characters etc.)
- observations are after the first row
- no missing values and there is no coding for missing values

## Read the .txt file

```
Drug = read.table("DrugTrial.txt",header = T)

##OR

Drug <- read.delim("C:/Users/jmorrison/OneDrive - The College of Wooster/College of Wooster/Fall2019/Da

head(Drug)
```

```
##   X.Group. X.Patient. X.Time. X.Baseline. X.Seizures.
## 1   'Drug'          1       1          15          11
## 2   'Drug'          2       1          13           6
## 3   'Drug'          3       1          12           8
## 4   'Drug'          4       1          18           4
## 5   'Drug'          5       1          30          15
## 6   'Drug'          6       1          14           7
```

```
str(Drug)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ X.Group.   : Factor w/ 2 levels "'Drug'","'Placebo'": 1 1 1 1 1 1 1 1 1 1 ...
##  $ X.Patient. : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ X.Time.    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ X.Baseline.: int  15 13 12 18 30 14 25 22 23 14 ...
##  $ X.Seizures.: int  11 6 8 4 15 7 12 21 17 2 ...
```

## Some information about the dataset

- effectiveness of a drug in reducing the number of epileptic seizures in patients
- two treatments: Drug and Placebo
- baseline seizure rates, as well as seizure rates for 5 months while on therapy

## ALWAYS check the imported data for missing values

- `NA` - missing values
- you can check for `NA` with `is.na(your_object)`

For Example

```
# Drug is the name of the dataframe
any(is.na(Drug))
```

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

```
#is.na(Drug)   #entrywise check

c<-data.frame(a=c(1,2), b=c(NA,1))
which(is.na(c))
```

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

This tells you if there are any missing values in the dataframe.

## How to view a specific variable

See the different levels of *Group*

- Method 1 - lists unique entries

```
Drug = read.table("DrugTrial.txt",header = T)
unique(Drug$Group)
```

```
## [1] Drug    Placebo
## Levels: Drug Placebo
```

- Method 2 - list the first entry of each unique group entry

```
Drug = read.table("DrugTrial.txt",header = T)
library(dplyr)
Drug %>% group_by(Group) %>% slice(1)
```

```
## # A tibble: 2 x 5
## # Groups:   Group [2]
##   Group   Patient  Time Baseline Seizures
##   <fct>     <int> <int>    <int>    <int>
## 1 Drug          1     1       15       11
## 2 Placebo       1     1       15       15
```

```
#Changing number in slice() changes the entry number
```

### Interested in reviewing the contents of the Drug dataset?

```
head(Drug)
```

```
##   Group Patient Time Baseline Seizures
## 1  Drug       1    1       15       11
## 2  Drug       2    1       13        6
## 3  Drug       3    1       12        8
## 4  Drug       4    1       18        4
## 5  Drug       5    1       30       15
## 6  Drug       6    1       14        7
```

```
dim(Drug)
```

```
## [1] 150   5
```

```
library(dplyr)
glimpse(Drug)
```

```
## Observations: 150
## Variables: 5
## $ Group    <fct> Drug, Drug, Drug, Drug, Drug, Drug, Drug, Drug, Drug,...
## $ Patient  <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1,...
## $ Time     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2,...
## $ Baseline <int> 15, 13, 12, 18, 30, 14, 25, 22, 23, 14, 15, 17, 26, 2...
## $ Seizures <int> 11, 6, 8, 4, 15, 7, 12, 21, 17, 2, 4, 8, 13, 2, 27, 1...
```

## Exporting Tables `write.table`

- prints your dataframe or matrix to a file or connection.
- the `write.table` function has similar parameters to `read.table`

Usage:

```
write.table(x, file = "", sep = " ",
           na = "NA", row.names = TRUE,
            col.names = TRUE)
```

- `?write.table` to get more information

## How do you save `Drug` as .csv file?

```
write.table(Drug, file = "First_Save_Drug.csv",
            row.names = FALSE,
            col.names = TRUE)
# Drug does not have row names
# You can also save to a .txt file here
```

## Another Way to save as .csv file `write.csv`

```
# by defalt write.csv separate entries by ","
write.csv(Drug, file = "Druggies2.csv", row.names = FALSE)
```

## Save as an Excel sheet?

```
#install.packages("writexl")
library(writexl)
 write_xlsx(Drug,path = "C:/Users/jmorrison/OneDrive - The College of Wooster/College of Wooster/Fall20
            ytics/Common Activities/Lectures/Lecture 2/Drugs.xlsx")
```

## Reading a .csv file using `read.csv`

Basic syntax:

```
read.csv(file, header = TRUE, sep = ",", ...)

read.csv2(file, header = TRUE, sep = ";", ...)
```

- Use `?read.csv` to get more information

read.csv and read.csv2 are identical to read.table except for the defaults. They are intended for reading 'comma separated value' files ('.csv') or (read.csv2) the variant used in countries that use a comma as decimal point and a semicolon as field separator.

### DataMalyria example - Some properties of the file

Will look at "dataMalyria.csv" - file from notes 1

NOTICE:

- First row - column names
- All the observations are numeric, characters or strings
- there appears to be no missing values

### Reading the .csv file

```
Mal = read.csv("dataMalyria.csv",header = TRUE)

#For more information about the dataset as imported
head(Mal)
```

```
##      country percent labels
## 1    Lesotho       0   <1%
## 2  Mauritius       0   <1%
## 3 Seychelles       0   <1%
## 4 Cabo Verde       0   <1%
## 5    Algeria       0   <1%
## 6      Egypt       0   <1%
```

```
str(Mal)   #Structure of dataframe
```

```
## 'data.frame':    53 obs. of  3 variables:
##  $ country: Factor w/ 53 levels "Algeria","Angola",..: 25 32 41 7 1 15 27 33 50 47 ...
##  $ percent: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ labels : Factor w/ 5 levels " <1% "," 1-4% ",..: 1 1 1 1 1 1 1 1 1 1 ...
```

### Another way to read the .csv file, but being specific!

Variable "country" as `character`:

```
Mal2 = read.csv("dataMalyria.csv",header = TRUE,
             colClasses=c("country"="character","percent"="numeric",
                          "labels"=NA))
#Looking at datset AGAIN!
head(Mal2)
```

```
##      country percent labels
## 1    Lesotho       0   <1%
## 2  Mauritius       0   <1%
## 3 Seychelles       0   <1%
## 4 Cabo Verde       0   <1%
## 5    Algeria       0   <1%
## 6      Egypt       0   <1%
```

```
str(Mal2)
```

```
## 'data.frame':    53 obs. of  3 variables:
##  $ country: chr  "Lesotho" "Mauritius" "Seychelles" "Cabo Verde" ...
##  $ percent: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ labels : Factor w/ 5 levels " <1% "," 1-4% ",..: 1 1 1 1 1 1 1 1 1 1 ...
```

## Reading the .csv file again…another way to be specific!

Variable "country" as `character`:

```
Mal3 = read.csv("dataMalyria.csv",header = TRUE,
             stringsAsFactors = F)

#Looking at the dataset once again!
head(Mal3)
```

```
##      country percent labels
## 1    Lesotho       0   <1%
## 2  Mauritius       0   <1%
## 3 Seychelles       0   <1%
## 4 Cabo Verde       0   <1%
## 5    Algeria       0   <1%
## 6      Egypt       0   <1%
```

```
str(Mal3)
```

```
## 'data.frame':    53 obs. of  3 variables:
##  $ country: chr  "Lesotho" "Mauritius" "Seychelles" "Cabo Verde" ...
##  $ percent: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ labels : chr  " <1% " " <1% " " <1% " " <1% " ...
```

## Using the save function

- `save` creates external representation of R objects to the specified file.
- `save.image()` is just a short-cut for 'save my current workspace'
- How to use it?

```
x<-c(1,2,3,4,5)
y<-c(2,4,3,6,7)

save(x,y, file = "fileName.RData")
```

## `load` saved data

- `load` reloads the Rdata we saved
- usage:

```
load("fileName.RData")
```

- `?load` for more information

### Save *Drug* data and load it

```
save(Drug,file="Druggy.RData")

load("Druggy.RData")
```

### Save workspace and load it

Why? your workspace has many objects.

```
getwd()   # get current working directory

save.image("myworkspace.RData")   #Saves workspace in path

load("myworkspace.RData")  # load can be used to load RData
```

Your workspace is everything you have been working on in the entire session.

Also remember that everything is saved in your current working directory! so `getwd()` checks your working directory

### Importing data from internet

- `read.table`, `read.csv` and `load`
- provide path to file or data via http link
- file name and location should always be a string
- specify parameters for a command if needed

### `read.table` and `http`

```
# file name and location should always be a string
Website = read.table(
  "http://courses.washington.edu/b517/Datasets/string.txt",
                   header=T)
head(Website)
```

```
##    x       y
## 1 10 34.7081
## 2 12 34.5034
## 3 14 36.5656
## 4 16 38.3125
## 5 18 42.5441
## 6 20 43.7210
```

## Import via RStudio

- Import non-R generated data: File > Import Dataset > ...
- Load R data: File > Open File > ...
- Caution: load saved workplace into current work space may overwrite existing objects

## View data in RStudio

In RStudio:

- go to *Environment* tab, which shows everything in the current `workspace`
- click on an object listed there
- the object chosen will be shown in a tab with the name being the object's name *the tab usually resides in the upper left corner of RStudio GUI