Intermediate R - Load Data And Environment - 1

One should look for what is and not what he thinks should be. – Albert Einstein

Load Data and Environment: Topic introduction

In this part of the course, we will cover the following concepts:

- Load data into R
- Manipulate variables within an R environment

Warm-up

- Think of the kinds of data that you encounter most commonly and consistently
- What **file extensions** do you typically associate with these datasets?
- What **file formats** are you most comfortable working with?
- Share your thoughts in the virtual chat

Module completion checklist

Objective	Complete
Read and write data in RStudio	
Load and clear variables in the R environment	

File Paths

- File paths are essentially directions that tell R where to import or export files on your computer
- Let’s say you want to go to the coffee shop to get breakfast
- How would you get there?
  - Start at home
  - Catch the downtown 6 train
  - Get out at Astor Place, facing south
  - Turn right at the corner of Astor Place
  - Walk 2 blocks
  - Turn right at E 4th street

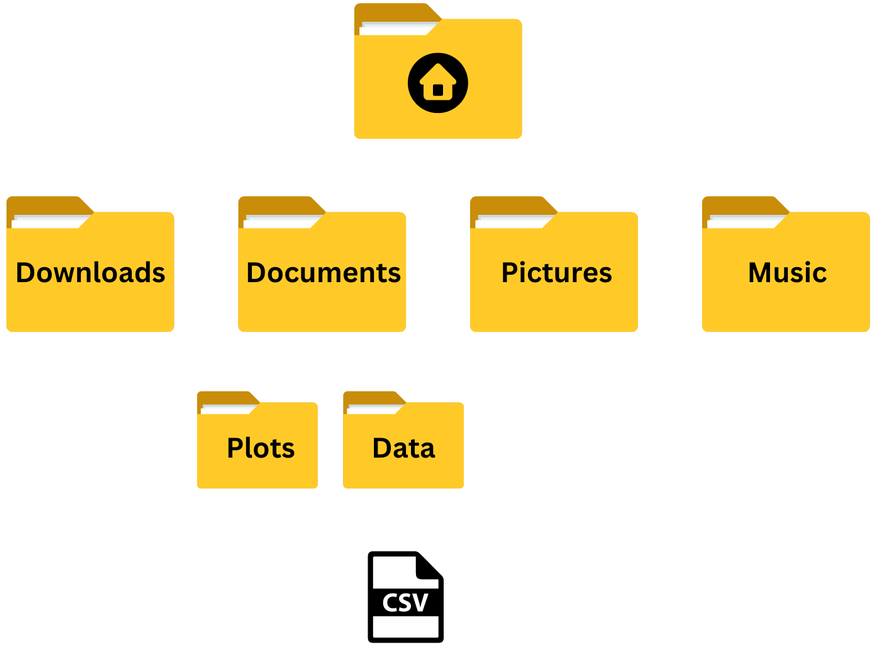
File Paths

- When setting file paths, you need have
  - specific instructions to get from the current to the desired location and
  - the correct sequence needed to arrive from the current to the desired location
- Consider this
  - How do you get the coffee shop if you are starting from work and not home?
  - What would happen if you were to catch the uptown 6 train?

Setting your path

- Instead of moving to and from geographic locations, locating a file on your computer involves moving from your home directory (or folder) to deeper, nested directories
- When writing path names,
  - begin with the home directory, which is often the username for the computer
  - separate each directory with a **backslash (/)** and **no spaces**
  - make sure it ends with the name of the desired file with the proper file extension
  - and encapsulate it within single or double quotations marks

Setting your path (contd)



- How would you write out the path to the CSV file?
- We would give R the following directions to the CSV
  - `"/Users/datasociety/Documents/Data/data.csv"` for Mac
  - `"C:/Users/datasociety/Documents/Data/data.csv"` for Windows

R’s working directory

- The **working directory** is the directory in which R will import or export files
- R has a default working directory, which can be found and set through RStudio’s `Global Options`

Setting the working directory via R

- We can use `getwd()` to get the working directory
- This is the current working environment that R is currently using to import and export files.

```
# Check the current working directory.
# Note: your directory path may look different.
getwd()
```

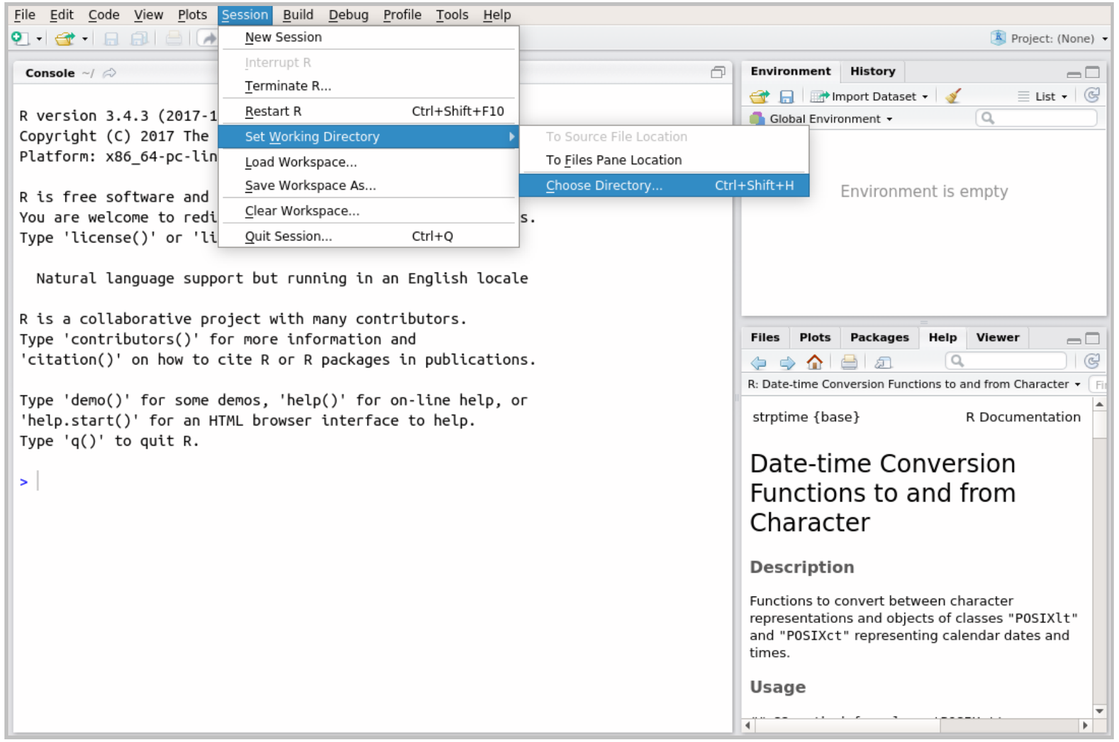
```
[1]
"/opt/atlassian/pipelines/agent/build/slides/I
```

- We can use `setwd()` to set or change the working directory
- We can pass a directory as a string to set or change the directory.

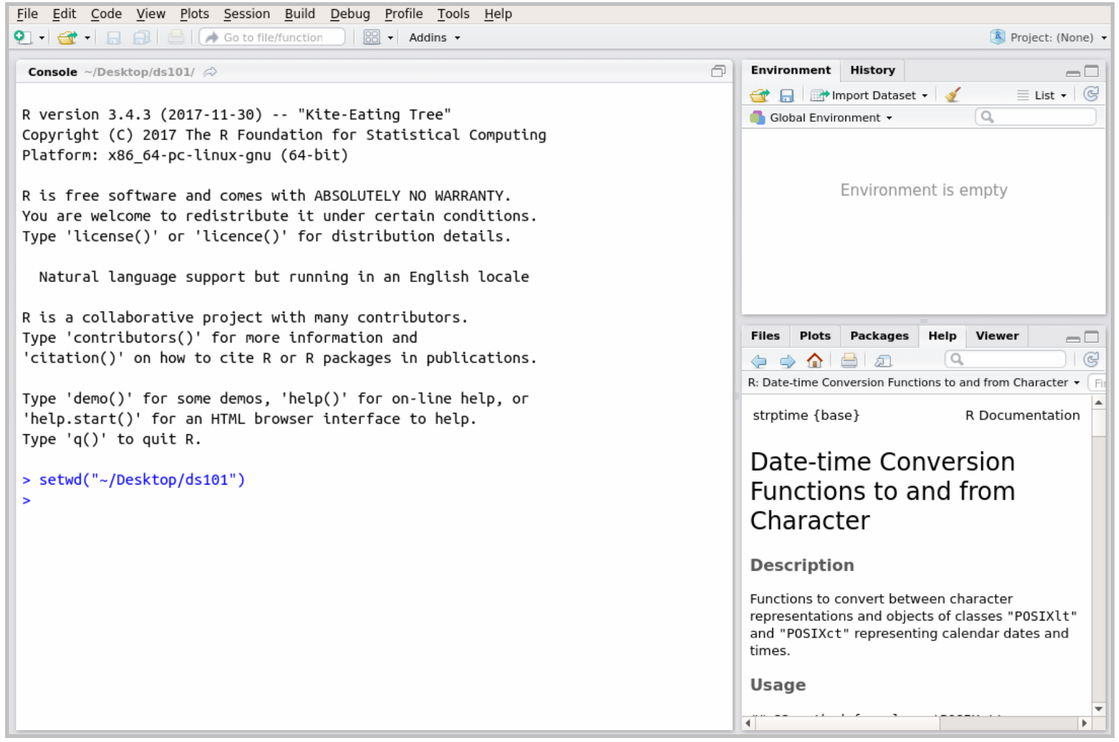
```
# Set the working directory.
# Note: your directory path may look different.
setwd()
```

Setting the working directory via GUI

- You can also set your working directory via RStudio's GUI

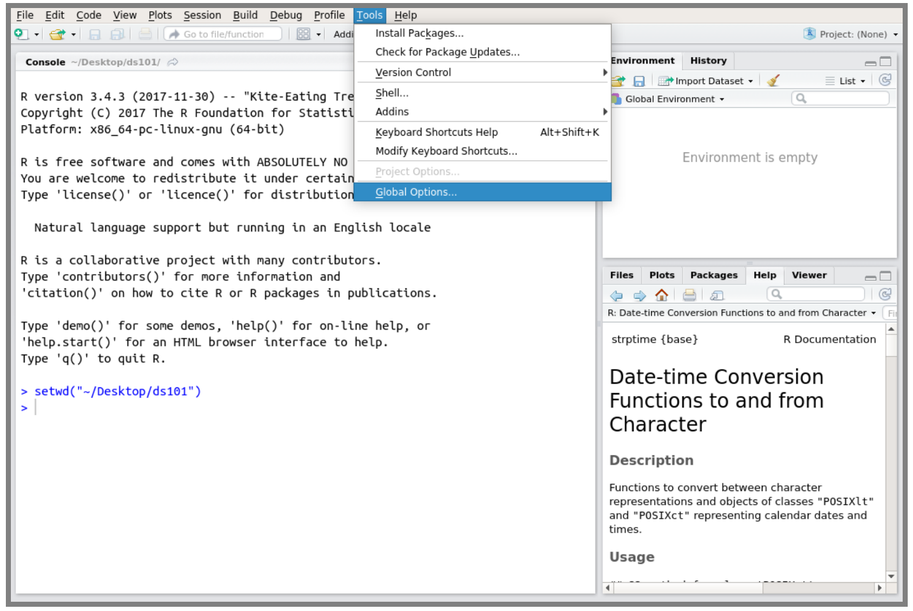


- Once the directory is set, you will see the command executed in the **Console**

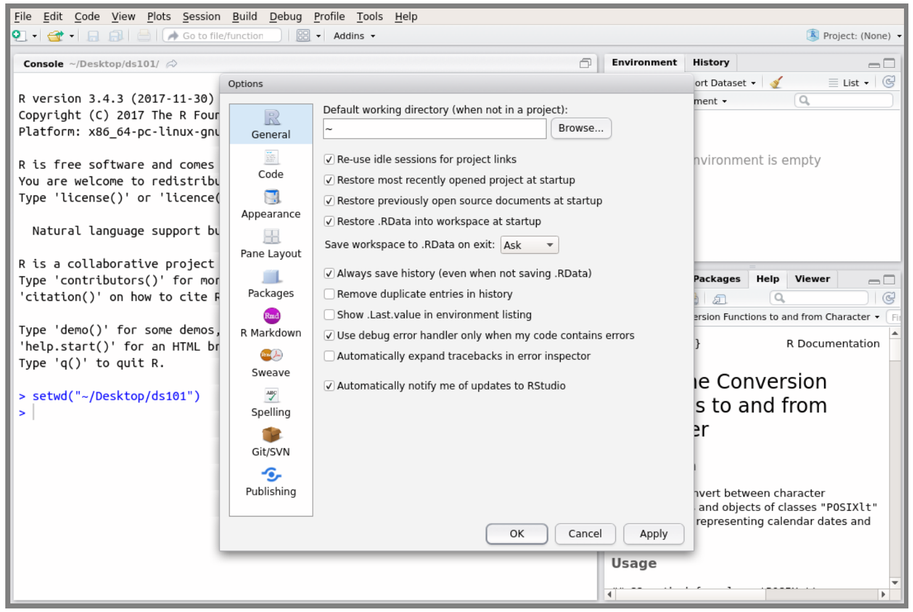


R’s default working directory

- Set a default working directory under **Tools > Global Options**



- Under the **General** section, check the current default working directory or change it using the **Browse** menu



Directory settings

- We can encode directory paths into variables and then change them without having to manually type the paths every time
- In order to maximize the efficiency of your workflow, you may want to use the `box` package and encode your directory structure into `variables`

```
install.packages("box")
```

- Let the `main_dir` be the variable corresponding to your materials folder
- Use `dirname()` to navigate through your computer's file hierarchy

```
# Set `main_dir` to the location of your materials folder.
# Note: your directory path may look different.

path = box::file()
main_dir = dirname(dirname(path))
```

Directory settings (cont'd)

- We will store all datasets in the `data` directory inside of the materials folder in your environment, so we'll save its path to a `data_dir` variable
- We will save all of the plots in the `plots` directory, corresponding to `plot_dir` variable

- To save some time retyping these paths, we can add a string to our `main_dir` variable by using `paste0` to add the extensions

```
# Make `data_dir` from the `main_dir` and
# remainder of the path to data directory.
data_dir = paste0(main_dir, "/data")
# Make `plots_dir` from the `main_dir` and
# remainder of the path to plots directory.
plot_dir = paste0(main_dir, "/plots")
```

- To list the files that start with a certain pattern, we use the `list.files()` function

```
list.files(path = data_dir, pattern="temp")
```

```
[1] "temp_heart_rate_subset.csv"
"temp_heart_rate.csv"
```

Loading dataset into R: read CSV files

- Most of the time, we work with data generated elsewhere that we need to load into our R environment
- R works with many different data types, but the most common one is `CSV`

```
# To read a C[omma] S[eparated] V[alues] file into
# R you can use a simple command `read.csv`.
temp_heart_data = read.csv(file = file.path(data_dir, "temp_heart_rate.csv"),  #<- provide file
path
                           header = TRUE,            #<- if file has header set to TRUE
                           stringsAsFactors = FALSE) #<- read strings as characters, not as factors
```

- In R, **factors** are variables with a limited number of different values, (i.e., categorical variables)
- Since we haven't profiled this dataset to know whether all strings should be converted to factors, we will leave `stringsAsFactors` set to `FALSE`

Viewing data in R

- First, we can take a general look into our dataset structure with `str()`

```
# Inspect the structure of the data.
str(temp_heart_data)
```

```
'data.frame':   130 obs. of  3 variables:
 $ Gender    : chr  "Male" "Male" "Male" "Male" ...
 $ Body.Temp : num  96.3 96.7 96.9 97 97.1 97.1 97.1 97.2 97.3 97.4 ...
 $ Heart.Rate: int  70 71 74 80 73 75 82 64 69 70 ...
```

- What other R functions can we use to explore our data?

Viewing data in R (cont'd)

- Then, we can inspect the head or tail of our data with `head()` or `tail()` function
- By default, `head()` will give you the **first six** rows and `tail()` will give you the **last six**
- However, you can also adjust the number of rows as the following example illustrates

```
head(temp_heart_data, 4) #<- Inspect the `head` (first 4 rows).
```

```
  Gender Body.Temp Heart.Rate
1   Male      96.3         70
2   Male      96.7         71
3   Male      96.9         74
4   Male      97.0         80
```

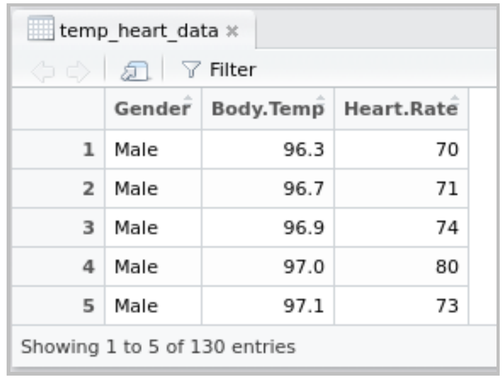
```
tail(temp_heart_data, 4) #<- Inspect the `tail` (last 4 rows).
```

```
    Gender Body.Temp Heart.Rate
127 Female      99.4         77
128 Female      99.9         79
129 Female     100.0         78
130 Female     100.8         77
```

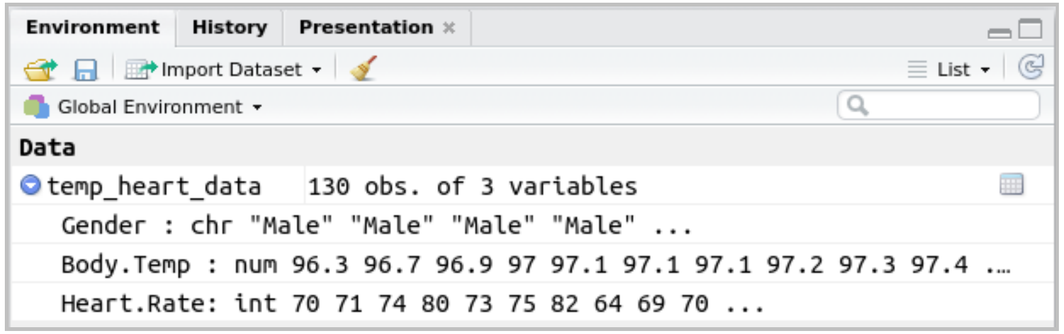
Viewing data in R (cont'd)

- Use `View` to see this data as a table in RStudio

```
View(temp_heart_data)
```



- You can also see the loaded data and variables in the `Environment` pane of RStudio



Other file types and commands in R

- Reading in other files is a matter of specifying their type

Command	File type
`read.csv("filename.csv")`	File with comma separated values
`read.table("filename")`	Tabulated data in a text file
`read.spss("filename.spss")`	File produced in SPSS
`read.dta("filename.dta")`	File produced in STATA
`read.ssd("filename.ssd")`	File produced in SAS
`read.JPEG("filename.jpg")`	Read JPEG image files

Saving data: write CSV files

- The most common way to share tabular data is by saving your data to a `CSV` file

```
# Let's save the first 10 rows of our data to a variable.
temp_heart_subset = temp_heart_data[1:10, ]
temp_heart_subset
```

```
   Gender Body.Temp Heart.Rate
1    Male      96.3         70
2    Male      96.7         71
3    Male      96.9         74
4    Male      97.0         80
5    Male      97.1         73
6    Male      97.1         75
7    Male      97.1         82
8    Male      97.2         64
9    Male      97.3         69
10   Male      97.4         70
```

```
# Write data to a CSV file providing 3 arguments:
write.csv(temp_heart_subset,              #<- name of variable to save
          file.path(data_dir, "temp_heart_rate_subset.csv"), #<- name of file where to save
          row.names = FALSE)              #<- logical value for row names
```

Module completion checklist

Objective	Complete
Read and write data in RStudio	✔
Load and clear variables in the R environment	

Clearing objects from environment

- Suppose we conclude a coding session and want to remove objects from the environment, or free up a variable name for reassignment

```
# List all objects in environment.
ls()
```

```
 [1] "conda_yaml"         "data_dir"           "env_name"
 [4] "main_dir"           "params"             "platform"
 [7] "plot_dir"           "session_info"       "temp_heart_data"
[10] "temp_heart_subset"
```

```
# Remove individual variable(s).
rm(X, x, this_is_a_valid_name, This.Is.Also.A.Valid.Name, unnamed_list) #<- example
rm(list=ls()) #<- actual command
```

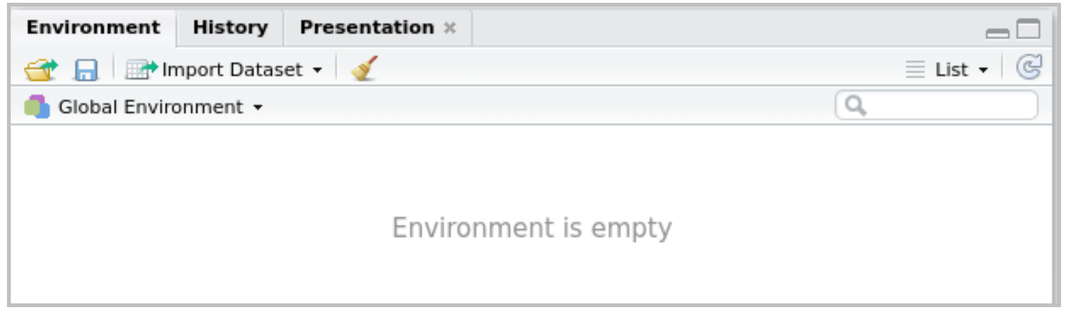
```
# List all objects again to check.
ls()
```

```
character(0)
```

- Notice the variables we have removed are gone

Clearing the entire environment

- A clear environment will always appear like this in the **Environment** pane



- You can also clear the environment by clicking on the **broom icon** at the top of the Environment pane

Module completion checklist

Objective	Complete
Read and write data in RStudio	✔
Load and clear variables in the R environment	✔

Knowledge check

Exercise

You are now ready to try Tasks 1-8 in the Exercise for this topic

Module completion checklist

Objective	Complete
Read and write data in RStudio	✔
Load and clear variables in the R environment	✔

Load Data and Environment: Topic summary

In this part of the course, we have covered:

- Load data into R
- Manipulate variables within R environment

Congratulations on completing this module!