Reading and Writing Files in R

Author: Cole Brookson **Date:** 13 June 2022

Quickly after beginning to use R, as a biologist it is likely you will want to analyze some data. These data will be often stored as external files in your computer, with the most common data file types being Excel Spreadsheet files (`.xls(x)`), Comma Separated Value files (`.csv`), or Plain Text (`.txt`) files.

Since it is best practice to **never** directly edit your raw data files (because what if you need to go back to the original version!?), a common workflow is to:

1. *Read your data into R* '
2. Perform some kind of analysis, creating new data objects, figures, or outputs, then
3. *Write out those data objects/figures/outputs to your computer for later use*

Steps 1 & 3 will be covered here.

There are, as there are with all things programming, multiple ways to perform these tasks in both base R and via packages.

Reading Files

When reading in a new file, after manually checking it and ensuring it is formatted in a readable way, note the path in your computer where the file is present. **NOTE** see section on workflow and the `here()` package.

Comma-Separated Value `.csv` Files

This is the recommended file type for reading data into R as it is flexible and easily transferred directly into a dataframe.

We can read in the file by assinging it directly to an object.

Base R

```
data <- read.csv("/path/to/file/file-name.csv", header = TRUE)
```

Here the first argument in the `read.csv()` function is the path pointing to the file in your computer. The second argument specifies `header = TRUE` which tells R that the first row of the `.csv` file will be the column names, not part of the data themselves.

Tidyverse

```
data <- readr::read_csv("/path/to/file/file-name.csv")
```

Excel Sheets `.xls(x)`

The best way to read in an excel sheet into R with with the `readxl` package. Almost identical to above:

```
data <- readxl::read_xls("/path/to/file/file-name.csv", sheet = "SHEETNAME")
```

Here, the path argument is identical above, but the `sheet =` argument is an optional argument that you can pass the name of a particular sheet in the file if you only want to read in one sheet out of many.

Plain Text `.txt`

Reading a text file is similar to the approaches to a csv file:

Base R

```
data <- read.delim("/path/to/file/file-name.csv")
```

This above option assumes that the file it tab-delimited.

Tidyverse

```
data <- readr::read_tsv("/path/to/file/file-name.csv")
```

Writing Files

Writing files is nearly identical to reading files.

Comma-Separated Value `.csv` Files

core R

```
write.csv(data_object, "/path/to/new/file/file-name.csv")
```

Tidyverse

```
readr::write_csv(data_object, "/path/to/new/file/file-name.csv")
```

Excel Sheets `.xls(x)`

Writing a file to an excel sheet is slightly unusual and wouldn't be recommended (`.csv` files are always better), but if you need to, there exists a number of packages to assist, the easiest of which being `writexl`.

```
library(writexl)
writexl::write_xlsx(data_object, "/path/to/new/file/file-name.xlsx")
```

Plain Text `.txt`

core R

```
write.table(data_object, "/path/to/new/file/file-name.txt")
```

Tidyverse

```
readr::write_delim(data_object, "/path/to/new/file/file-name.txt")
```

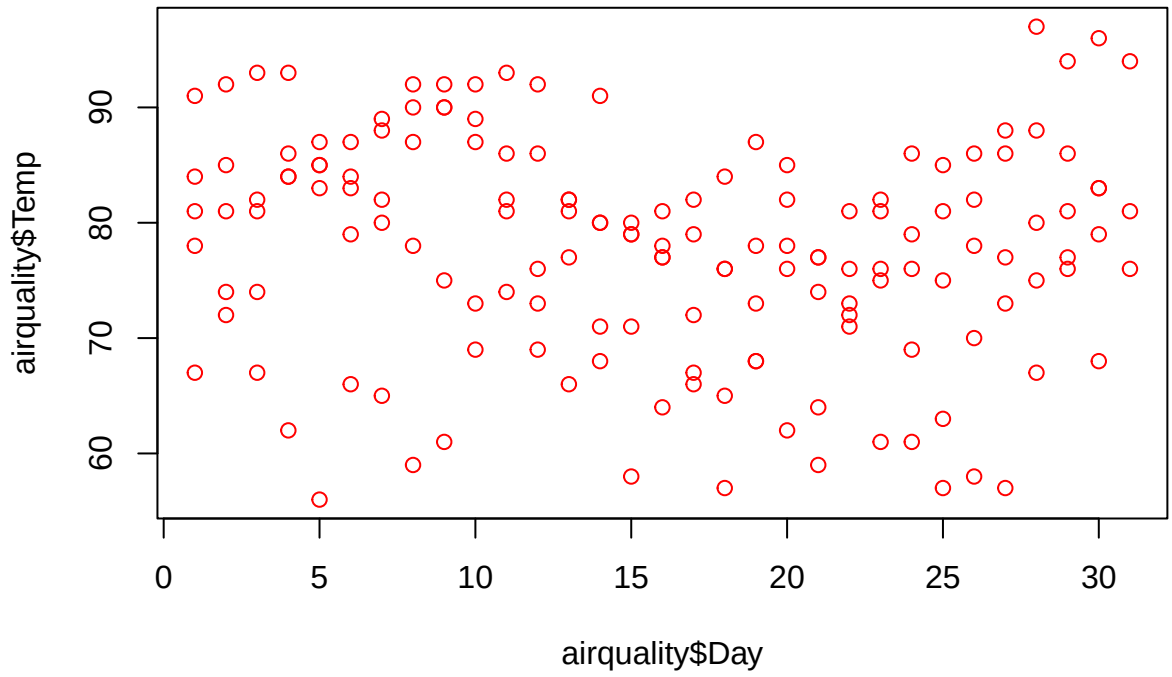
Writing Figures

It is common to create a figure in R and want to save it to the local machine. If you are plotting in `ggplot2`, then there are specific functions for those objects, but if you are plotting in base R, you will use built-in functions to save as whatever file type you want.

Using the example built-in dataset `airquality`, we can make an example plot to save.

Saving an image created in base R

```
plot(x = airquality$Day, y = airquality$Temp, col = "red")
```



If this is the plot we wish to save, we can save it as any common figure file type (e.g. `.png`, `.jpg`, `.tif`, `.pdf` etc.)

```
# first use the command for the file type
png(file="/path/to/plot/saving_plot.png")
# then we create the plot
hist(Temperature, col="darkgreen")
# this command saves the file and closes the connection to the file
dev.off()
```

Saving a ggplot2 object

It is best to use the let of functions from `ggplot2` to save these plots.

```
ggplot2::ggsave("/path/to/plot/saving_plot.png",
                plot_object)
```

Note here that we can change the type of file that gets saved by simply changing the extension on the file name.