Data Output

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Data Output

While its nice to be able to read in a variety of data formats, it's equally important to be able to output data somewhere.

The `readr` package in the tidyverse provides data exporting functions which have the pattern `write_*`:

- `write_csv()`,
- `write_delim()`, others.

From `write_csv()` documentation:

```
write_csv(x, file,
  na = "NA", append = FALSE,
  col_names = !append, quote_escape = "double",
  eol = "\n", path = deprecated()
)
```

```
Rows: 9794 Columns: 31
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (24): LocationAbbr, LocationDesc, TopicType, TopicDesc, MeasureDesc, Dat...
dbl  (7): YEAR, Data_Value, Data_Value_Std_Err, Low_Confidence_Limit, High_C...
```

```
ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```


Data Output

`x`: data frame you want to write

`file`: file path where you want to R object written; it can be:

- an absolute path,
- a relative path (relative to your working directory),
- a file name only (which writes the file to your working directory)
- remember to include the file extension (`.csv`, `.txt`, or `.tsv`)

```
# Examples

write_csv(dat, file = "YouthTobacco_newNames.csv")

write_delim(dat, file = "YouthTobacco_newNames.csv", delim = ",")
```

GUT CHECK!

What does `write_csv()` do? Saves data to:

A. R's memory

B. A file on your hard drive

C. A ggplot

R binary file

`.rds` is an extension for R native file format.

`write_rds()` and `read_rds()` from `readr` package can be used to write/read a single R object to/from file.

Saving datasets in `.rds` format can save time if you have to read it back in later.

```
# write an object: a data frame "dat"
write_rds(dat, file = "yts_dataset.rds")

# write an object: vector "x"
x <- c(1, 3, 3)
write_rds(x, file = "my_vector.rds")

# read an object from file and assign to a new object named "y"
x2 <- read_rds(file = "my_vector.rds")
x2
```

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

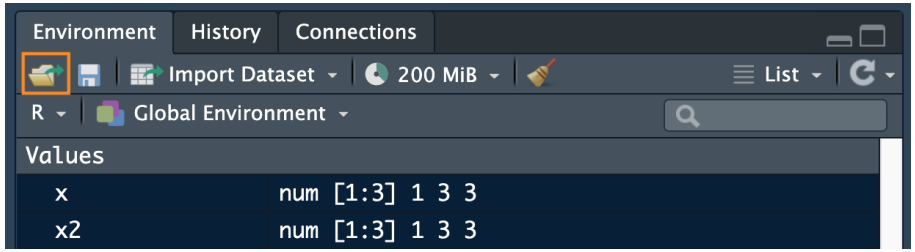

Saving multiple objects

You may want to export a set of objects from R for later use, either to save time or to use in another R script. You can output these to an `.RData` file individually, or save your entire Environment with `save.image()`.

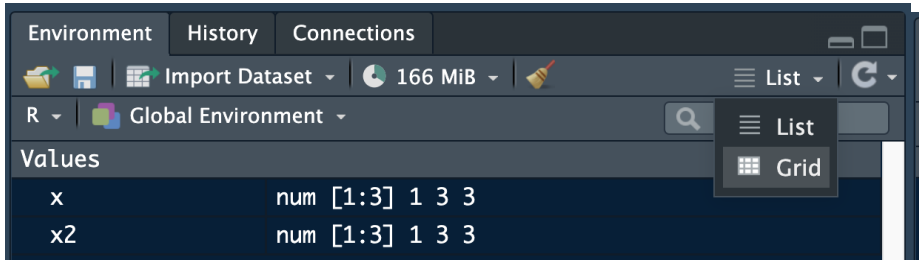
```
save(x, x2, file = "x_x2_output.RData")
save.image(file = "my_environment.RData")
```

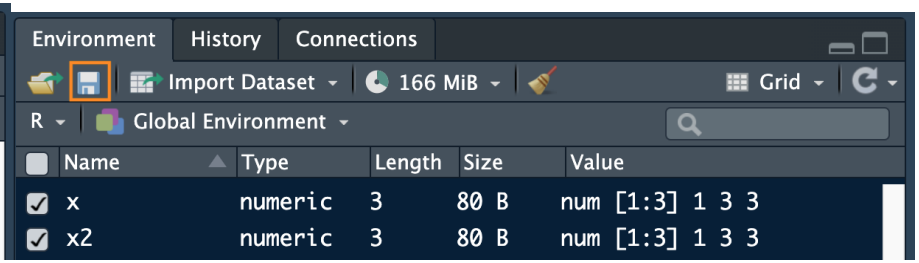
Using RStudio for importing/exporting data

If there is an `.rds` or `.RData` file that you want to work with, you can open it into your environment using the file icon.



Can also save your entire environment or a subset of objects in your environment to a new `.RData` file with the save icon. Click the "List" button and switch to "Grid" to select which objects to delete or keep before saving the Environment.





REMINDER: Saving a ggplot to file

A few options:

- RStudio > Plots > Export > Save as image / Save as PDF
- RStudio > Plots > Zoom > [right mouse click on the plot] > Save image as
- In the code

```
ggsave(filename = "saved_plot.png",  # will save in working directory
       plot = rp_fac_plot,
       width = 6, height = 3.5)              # by default in inches
```

Summary

- Use `write_csv()` and `write_delim()` from the `readr` package to write your (modified) data
- `.rds` files can be handy for saving intermediate work
- Can save environment (or subset) using `save()` and `save.image()`

Resources & Lab

- Class Website
- Data Output Lab
- Posit's Data Import Cheatsheet
- Day 2 Cheatsheet

Image by Gerd Altmann from Pixabay