

GAPS R Workshop 2024

Russell Glenn (rglenn3@uwo.ca)

Hunter Driggers (mdrigger@uwo.ca)

William Poirier (wpoirier@uwo.ca)

Justine Bechard (jbecha@uwo.ca)

Workshop Overview

Goal: Get you familiar with R so you can focus on the material of Methods 1 and not your software program.

10:00 - 12:00	Getting to Know R (Russell and William)
12:00 - 1:00	Lunch
1:00 - 3:00	Starting to Work in R (Hunter and Justine)

Ask questions! When not presenting, we're roaming the room to help.

Session 1: Getting to Know R

1. What is R? Why do we use it? What can it do?
2. A note on coding
3. Downloading R and R Studio
4. Introduction to the R interface
5. Running code

Acknowledgements / Other Resources

I borrowed heavily from Dr. Charles Lanfear's excellent [introductory R workshop](#)

Other resources used include:

[YaRrr! The Pirate's Guide to R](#) (free introductory book)

[Teacup Giraffes Intro to R](#) (free introductory R exercises)

["R For Data Science"](#) by Hadley Wickham (available free online)

Check these out!

What is R?

R is a programming language for statistical computing.

It might be useful to think of R as two things:

1. A programming language
2. A program you run on your computer

R is the “tool” you use to work on your “raw materials” (dataset) to find the answer to your research question.

What is R?

R allows us to do things like:

- Wrangle data

- Perform statistical computations

- Visualize data

- Scrape the web

- Get a good grade in Methods 1

- And many other things!

Why use R (and not Stata, SPSS, etc.)?

R is free

R has a large community that develops packages and gives support

R can handle any data format

R makes replication easy

R is a language, so it can do more

R is similar to other programming languages

A Note on Coding

We talk to other humans using languages like English or French.

We talk to *computers* (including R) using *code*.

Think of code as a language that lets you give R very specific instructions.

Working in R is essentially using code to instruct R to do a series of things.

A Note on Coding

Think of code as a language that lets you give R very specific instructions.

In English: “Tell me how many rows are in the 2019 Canadian Election Study (CES).”

In R: `nrow(ces2019)`

In English: “Keep only Conservative voters from the 2019 CES.”

In R: `ces2019 %>% filter(voteChoice == conservative)`

A Note on Coding

Coding can often feel intimidating as you get started.

But don't worry!

1. *Everybody* has been there - it's just part of the process.
2. *Everybody* in this room is completely capable of learning to code
3. Many people start off hating coding and come to love it

As my Methods 1 prof used to say, success at coding is “0% intelligence and 100% effort and practice.”

A Note on Coding

Don't be discouraged if you get lost during this workshop (we're going fast).

Do your best, and think about it as a chance to get exposed to R and what it can do.

We're here to help, so please ask!

Downloading R

Before doing anything, we need to download R to our computers.

Go to CRAN: <https://cloud.r-project.org>

- The Comprehensive R Archive Network.
- CRAN is composed of a set of mirror servers distributed around the world and is used to distribute R and R packages.

Downloading R

Windows

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux \(Debian, Fedora/Redhat, Ubuntu\)](#)
- [Download R for macOS](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

R for Windows

Subdirectories:

[base](#)
[contrib](#)
[old contrib](#)
[Rtools](#)

Binaries for base distribution. This is what you want to [install R for the first time](#).

Binaries of contributed CRAN packages (for $R \geq 4.0.x$).

Binaries of contributed CRAN packages for outdated versions of R (for $R < 4.0.x$).

Tools to build R and R packages. This is what you want to build your own packages on Windows, or to build R itself.

R-4.4.1 for Windows

[Download R-4.4.1 for Windows](#) (82 megabytes, 64 bit)

[README on the Windows binary distribution](#)
[New features in this version](#)

This build requires UCRT which is part of Windows since Windows 10 and Windows Server 2016. On older systems, UCRT has to be installed manually from [here](#)

Mac

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux \(Debian, Fedora/Redhat, Ubuntu\)](#)
- [Download R for macOS](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Latest release:

R 4.4.1 binary for macOS 11 (**Big Sur**) and higher, signed and notarized packages.

Contains R 4.4.1 framework, R.app GUI 1.80, Tcl/Tk 8.6.12 X11 libraries and Texinfo 6.8. The latter two components are optional and can be omitted when choosing "custom install", they are only needed if you want to use the tcltk R package or build package documentation from sources.

macOS Ventura users: there is a known bug in Ventura preventing installations from some locations without a prompt. If the installation fails, move the downloaded file away from the *Downloads* folder (e.g., to your home or Desktop).

For Apple silicon (M1-3) Macs:

[R-4.4.1-arm64.pkg](#)

SHA1-hash: 6165608170970b0dd881449ed92d098e52204830
(ca. 94MB, notarized and signed)

For older Intel Macs:

[R-4.4.1-x86_64.pkg](#)

SHA1-hash: e56eb09244121d7db78b41d3c06a737969365
(ca. 96MB, notarized and signed)

Downloading R Studio

Now that R is downloaded, we will download **R Studio**.

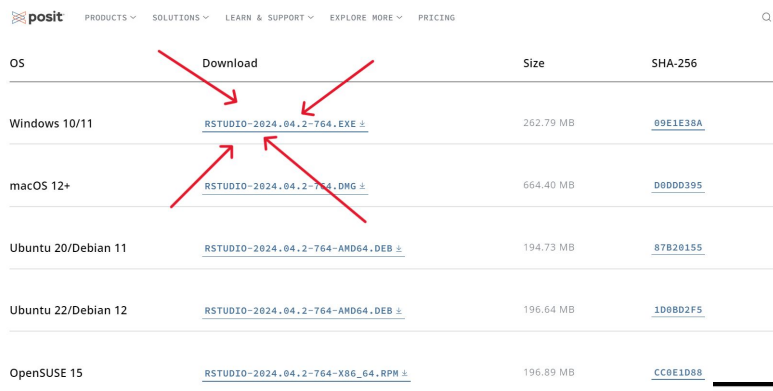
R Studio is an Integrated Development Environment (IDE) for programming in R

What you need to know: R Studio is where you will actually do your work in R

Downloading R Studio

Windows

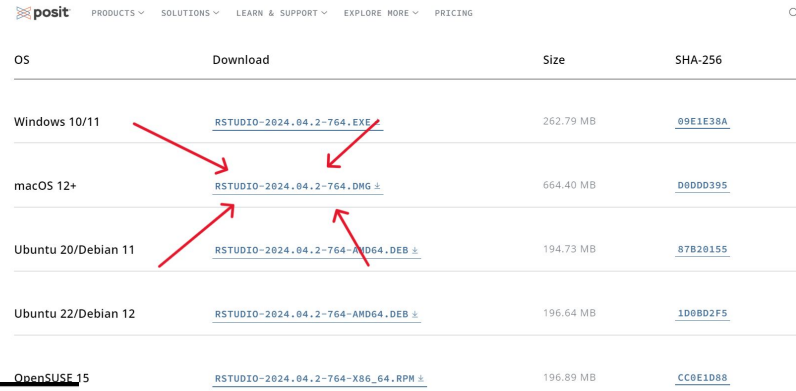
<http://www.rstudio.com/download>



OS	Download	Size	SHA-256
Windows 10/11	RSTUDIO-2024.04.2-764.EXE ±	262.79 MB	09E1E38A
macOS 12+	RSTUDIO-2024.04.2-764.DMG ±	664.40 MB	D0DD0395
Ubuntu 20/Debian 11	RSTUDIO-2024.04.2-764-AMD64.DEB ±	194.73 MB	87B20155
Ubuntu 22/Debian 12	RSTUDIO-2024.04.2-764-AMD64.DEB ±	196.64 MB	108BD2F5
OpenSUSE 15	RSTUDIO-2024.04.2-764-X86_64.RPM ±	196.89 MB	CC0E1D88

Mac

<http://www.rstudio.com/download>



OS	Download	Size	SHA-256
Windows 10/11	RSTUDIO-2024.04.2-764.EXE ±	262.79 MB	09E1E38A
macOS 12+	RSTUDIO-2024.04.2-764.DMG ±	664.40 MB	D0DD0395
Ubuntu 20/Debian 11	RSTUDIO-2024.04.2-764-AMD64.DEB ±	194.73 MB	87B20155
Ubuntu 22/Debian 12	RSTUDIO-2024.04.2-764-AMD64.DEB ±	196.64 MB	108BD2F5
OpenSUSE 15	RSTUDIO-2024.04.2-764-X86_64.RPM ±	196.89 MB	CC0E1D88



Introducing the R Studio Interface

At this point I'll move to R studio, but you can use the following slides for reference.

Open R Studio

File > New File > R Script

Introducing the R Studio Interface

Top Left: Source pane

- Where you create and edit **R scripts**
- Where you view datasets
- Browse with tabs

Bottom Left: Console

- Where R actually evaluates code
- Code you execute from the script is sent to the console

Running Code Demonstration

Type

```
print("Hello world!")
```

Into the script (top left pane).

To execute, ensure your cursor is on the same line as the code you just typed and press

```
ctrl + enter (PC)
```

```
command + return (Mac)
```

Running Code Demonstration

Look at your **console** (bottom left)

The line of code you just wrote in the **script**...

```
> print("Hello world!")
```

...appears in the **console**.

The **output** of your code also appears in your console. In this case,

```
[1] "Hello world!"
```

The [1] is an **index** - don't worry about it for now, but keep it in mind for later.

Running Code Demonstration

There are a few ways to run code:

1. When the cursor is on a line of code, press `ctrl + enter` (command + return)
2. Highlight the code you want to run and press `ctrl + enter` (command + return)
3. You can also click the green `run` button at the top of the pane instead of using the keyboard shortcuts.

Running Code Demonstration

You can also type code directly into the **console** and press `enter` to execute it.

Let's try running the same line of code in the **console**.

Type

```
print("Hello world!")
```

into the **console** (bottom left).

Press `enter`.

Notice that we get the same output as before!

Running Code Demonstration

It's always better to code in the **R script** than the console.

When you execute code from the **console**, it disappears.

If you want to run the same code again, or if you make a mistake and get an error, you need to type out the whole line of code again.

Since the code you type in the **script** doesn't disappear, it avoids this problem!

Exercise

(1) Modify the code you just learned to **print your name**.

- a. From the **script**
- b. From the **console**

Hint: Be sure that the brackets and quotation marks are in the right place!

(2) Use R to solve this math problem: $2893 - 787$

Introducing the R Studio Interface

Top Right: The Environment / History Panel

- Environment tab lists the objects in your environment.
- Code history tab shows the code that you've executed

Introducing the R Studio Interface

Bottom Right: tabs for

- Browsing files on your hard drive
- Viewing plots,
- Managing packages,
- Viewing help files
 - Search for topics in the search bar (e.g., we can look up the “print” function we used earlier)