Introduction to R and RStudio

Session outline

1. What is R?
2. Why should we use R?
3. R and RStudio
4. Packages
5. Using the console
6. Scripts

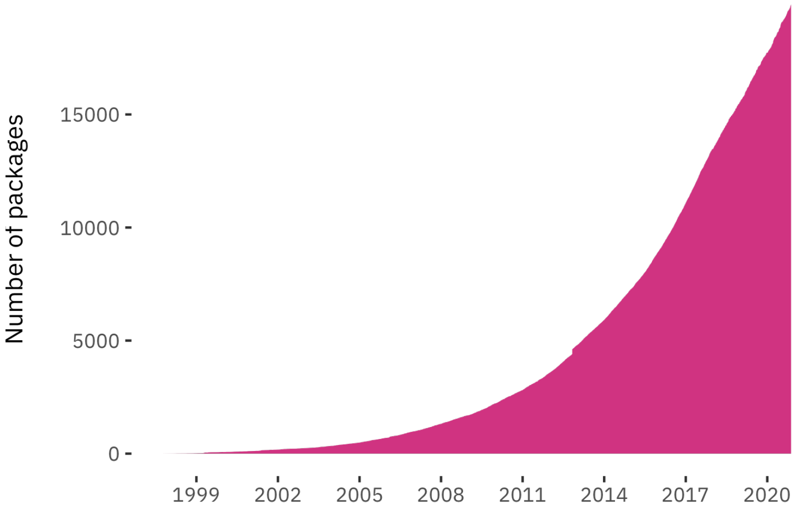
What is R?

- R is a free, open source language for statistical computing.

1992	Development began as research project in Auckland, by Robert Gentleman and Ross Ihaka, based on the S (1976) and S-PLUS (1988) languages.
1993	First release of R
2000	Version 1.0 released
2004	First UseR! conference in Vienna, Austria
2013	Version 3.0 released
2015	R Consortium founded
2017	10,000 packages published on CRAN
2020	Version 4.0 released

- R is one of the fastest growing programming languages, especially for statistics and data science.
- A large (and growing) ecosystem, with over 18,000 packages.

There are over 18,000 packages for R



https://gist.github.com/daroczig/3cf06d6db4be2bbe3368

Why use R?

Strengths

- Free and open source
- Big (and friendly) user community
- Excellent resources
- Very flexible
- Over 18,000 packages
- Rapid development
- Industry adoption
- It's a programming language.

Why use R?

Strengths

- Free and open source
- Big (and friendly) user community
- Excellent resources
- Very flexible
- Over 18,000 packages
- Rapid development
- Industry adoption
- It's a programming language.

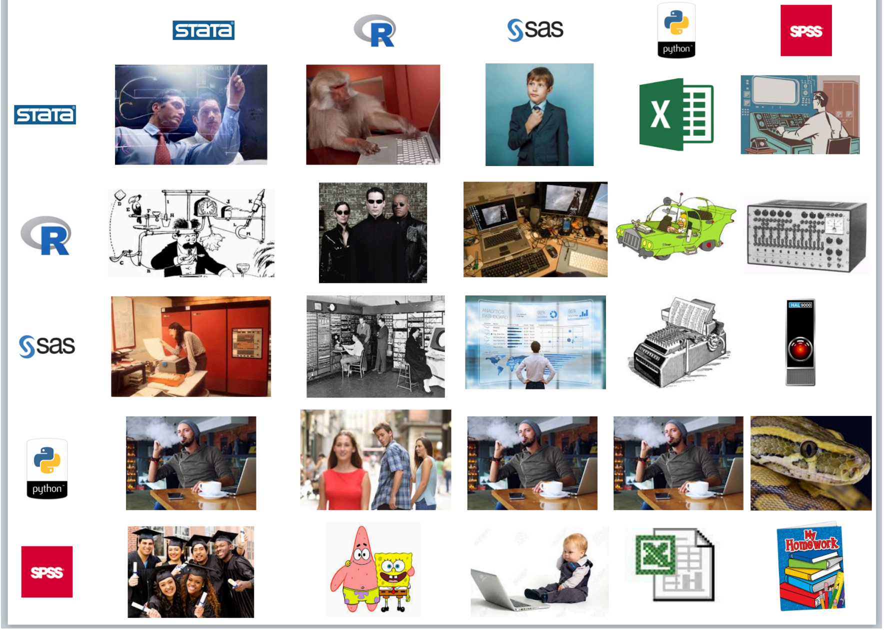
Weaknesses

- 18,000 packages. Which should I use?
- Rapid development
- Package management
- R is slow?
    - R used to be slow; less true today
    - Many functions now written in `C++` via `Rcpp`
- It's a programming language.

	R	Python	Julia	Stata	C++
Licence	Open	Open	Open	Closed	Open
Version 1.0	2000	1994	2018	1985	1985
Easy of use*	Moderate	Hard	Moderate	Easy	Very hard
Extensibility	Excellent	Good	Good	Limited	Very limited
Industry support	Excellent	Excellent	Limited	Limited	Good
Performance	OK	Good	Very good	Good	Excellent
Execution	Interpreted	Interpreted	JIT	Interpreted	Compiled

*For statistical computing.

AS SEEN BY USERS OF ...



Source: `https://kieranhealy.org/blog/archives/2019/02/07/statswars/`

Should I learn R or Python?

RStudio

Installing R and RStudio

R is a language.

RStudio is an environment for working with R.

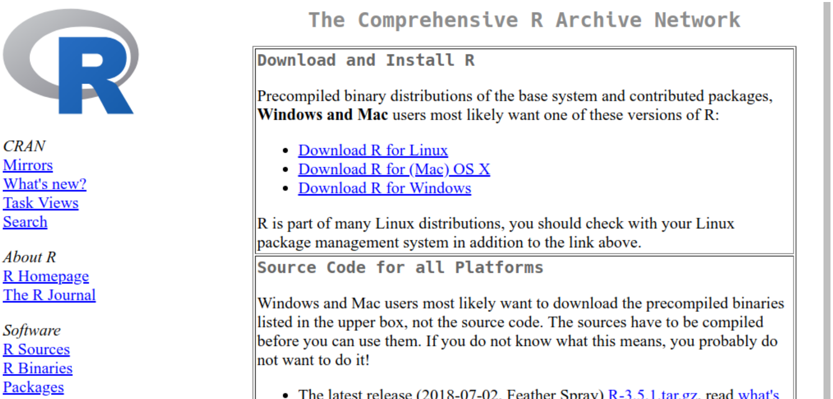
Installing R and RStudio

R is a language.

RStudio is an environment for working with R.

Installing R

Installing RStudio



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages

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
- Download R for (Mac) OS X
- 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.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2018-07-02, Feather Spray) R-3.5.1.tar.gz, read what's

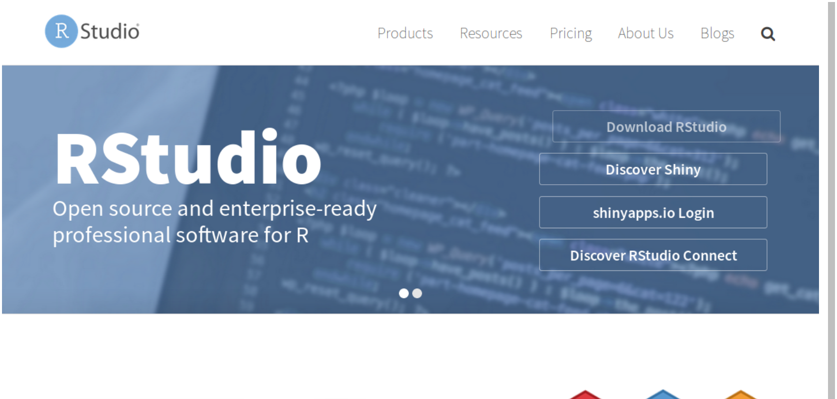
Installing R and RStudio

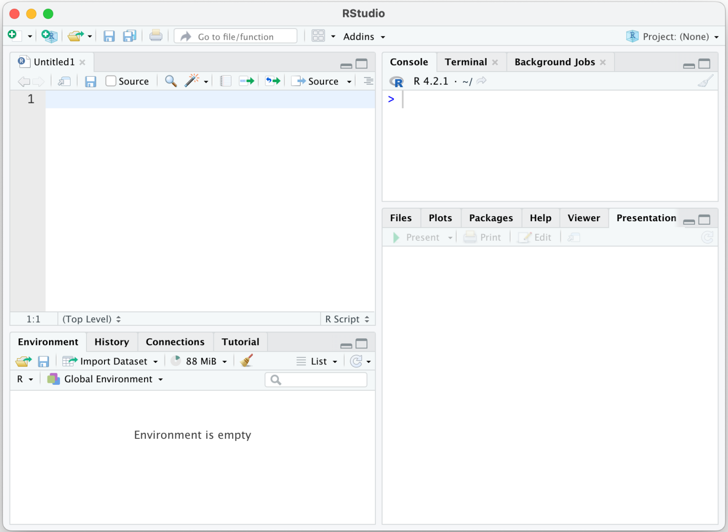
R is a language.

RStudio is an environment for working with R.

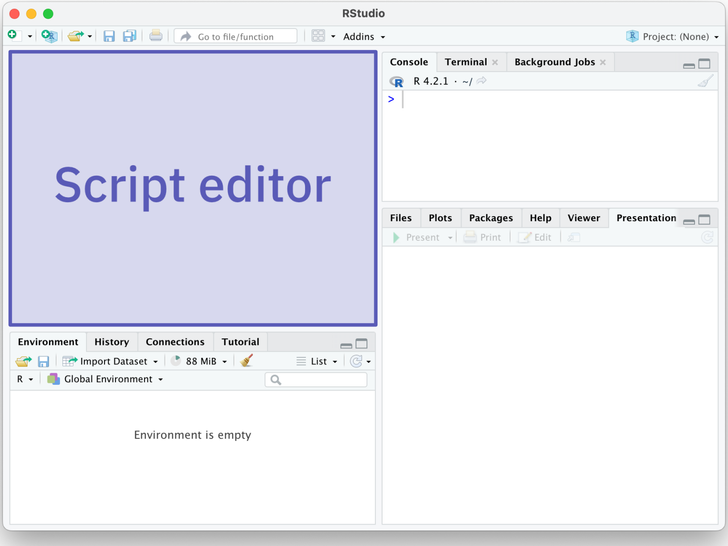
Installing R

Installing RStudio



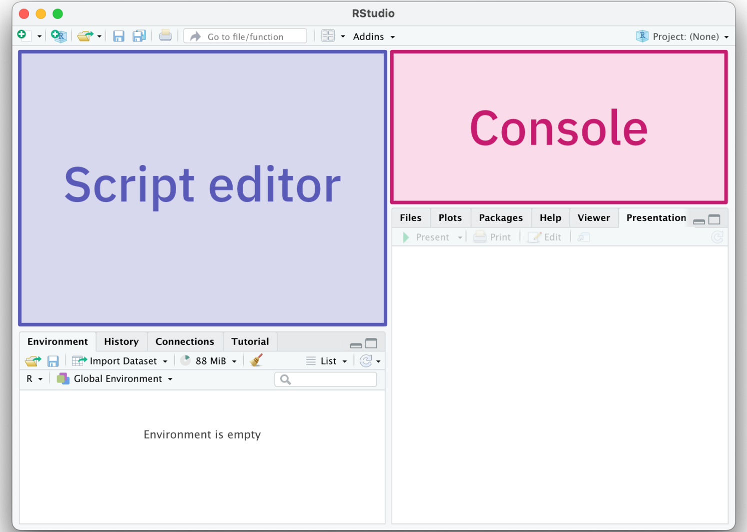


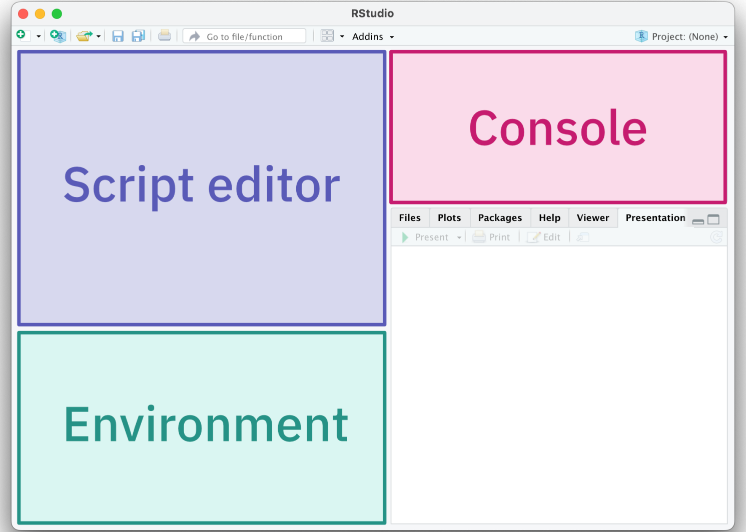
Environment is empty

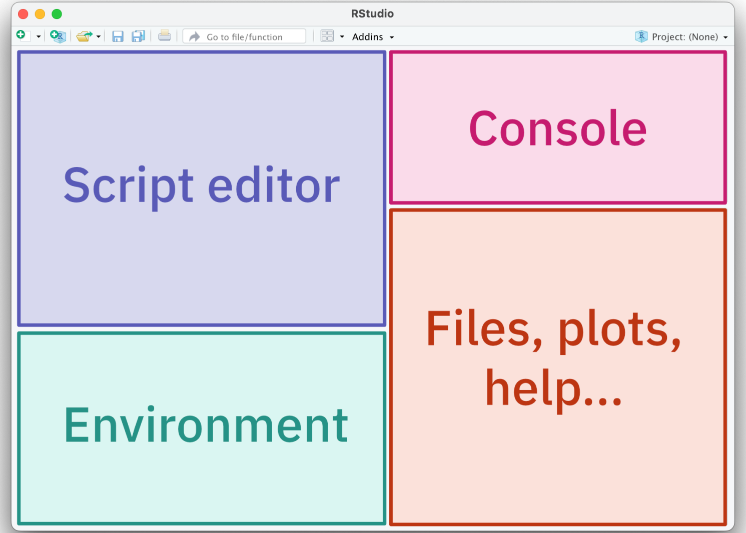


```
R 4.2.1 · ~/
>
```

Environment is empty

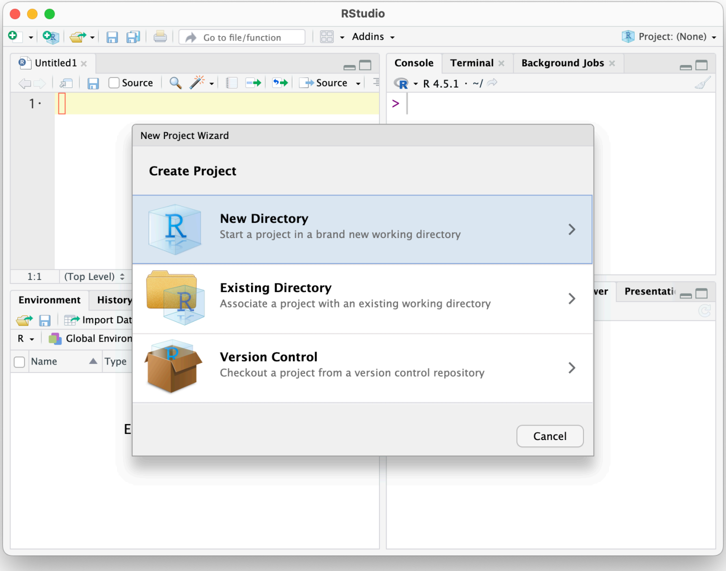

RStudio
Go to file/function
Addins
Project: (None)
Script editor
Console
Files
Plots
Packages
Help
Viewer
Presentation
Present
Print
Edit
Environment
History
Connections
Tutorial
Import Dataset
88 MiB
List
R
Global Environment
Environment is empty


RStudio
Go to file/function
Addins
Project: (None)
Script editor
Console
Environment
Files
Plots
Packages
Help
Viewer
Presentation
Present
Print
Edit



RStudio
Go to file/function
Addins
Project: (None)
Script editor
Console
Environment
Files, plots, help...

Projects in RStudio



New Project Wizard

Create Project

New Directory
Start a project in a brand new working directory

Existing Directory
Associate a project with an existing working directory

Version Control
Checkout a project from a version control repository

Cancel

R packages

R packages

We use packages to add functionality to R.

> *A collection of scripts designed to implement a particular function or method.*

They represent one of the principle strengths of R: if a method exists, someone has probably written a package.

R comes with many packages out-of-the-box, such as `datasets`, `graphics`, and `stats`. These are referred to as 'base' packages.

Installing packages

We can install packages by typing:

```
install.packages("lme4")
```

To load this package, we'd then type:

```
library(lme4)
```

(Note the use of quotation marks.)

You should install packages once. But you need to load them every time you use them.

How to choose which packages to use?

My preferences:

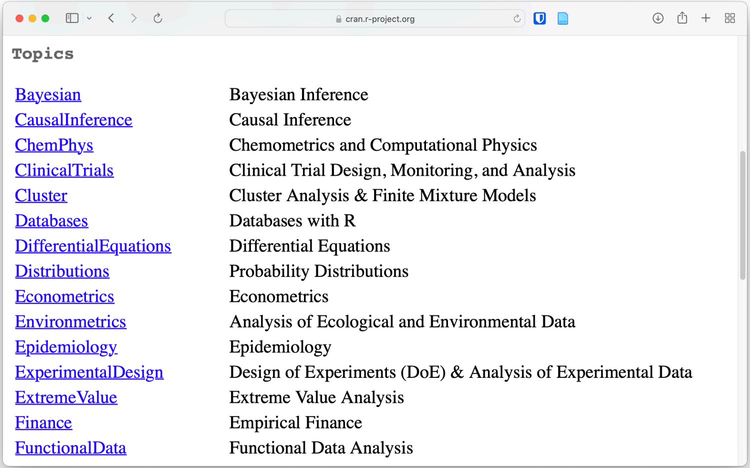
1. Use as few packages as possible (but as many as needed).
2. Use big general packages (rather than small specialised ones).
3. Use packages that have been around for a long time and are regularly updated.
4. Use stable, reliable packages.
5. At times, ignore all these rules.

Your preferences may differ!

> For example: `ggplot2`, `dplyr`, `tidyr`, `data.table`, `lme4`

Use the CRAN Task Views

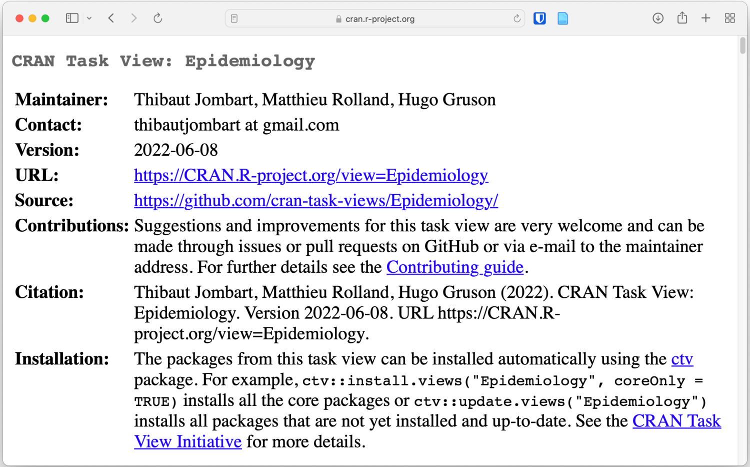
`cran.r-project.org/web/views`



Topics

Bayesian	Bayesian Inference
CausalInference	Causal Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
Databases	Databases with R
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
Epidemiology	Epidemiology
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis

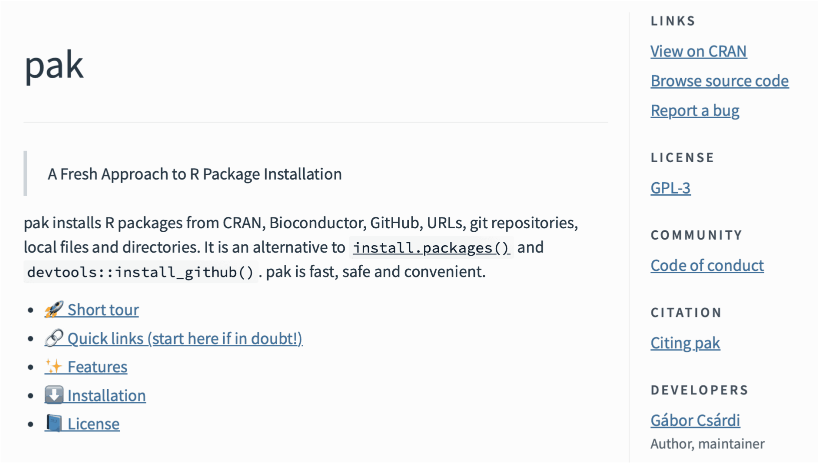
cran.r-project.org/web/views/Epidemiology



CRAN Task View: Epidemiology

Maintainer:	Thibaut Jombart, Matthieu Rolland, Hugo Gruson
Contact:	thibautjombart at gmail.com
Version:	2022-06-08
URL:	https://CRAN.R-project.org/view=Epidemiology
Source:	https://github.com/cran-task-views/Epidemiology/
Contributions:	Suggestions and improvements for this task view are very welcome and can be made through issues or pull requests on GitHub or via e-mail to the maintainer address. For further details see the Contributing guide.
Citation:	Thibaut Jombart, Matthieu Rolland, Hugo Gruson (2022). CRAN Task View: Epidemiology. Version 2022-06-08. URL https://CRAN.R-project.org/view=Epidemiology.
Installation:	The packages from this task view can be installed automatically using the ctv package. For example, `ctv::install.views("Epidemiology", coreOnly = TRUE)` installs all the core packages or `ctv::update.views("Epidemiology")` installs all packages that are not yet installed and up-to-date. See the CRAN Task View Initiative for more details.

Tip: Install packages with pak



pak

> A Fresh Approach to R Package Installation

pak installs R packages from CRAN, Bioconductor, GitHub, URLs, git repositories, local files and directories. It is an alternative to `install.packages()` and `devtools::install_github()` . pak is fast, safe and convenient.

- 🚀 Short tour
- 🔗 Quick links (start here if in doubt!)
- ✨ Features
- ⬇️ Installation
- 📘 License

LINKS

View on CRAN

Browse source code

Report a bug

LICENSE

GPL-3

COMMUNITY

Code of conduct

CITATION

Citing pak

DEVELOPERS

Gábor Csárdi

Author, maintainer

pak.r-lib.org

Scripts

Scripts are plain text files containing R code

Similar to Stata do-files or SPSS syntax files.

```r
# Title:       Example script
# Author:      Author's name
# Date:        Today's date
# Description: This script...

library(tidyverse)
library(here)
library(lme4)

# Import raw data
raw <- read.dta("source_data.dta")

# Fit linear model
model_1 <- lm(y ~ x1 + x2, data = raw)
```

Header

Load packages

Comments

Scripts (cont.)

- You should annotate your scripts with comments.

  ```
  # This is a comment.
  ```

  What makes a good comment?

- Save your scripts with the `.R` file extension

  e.g., `1-Data-prep.R`.

- Install packages once, load them every time you use them. Place at the top of your script.

  ```
  library(tidyverse)
  library(here)
  ```

Scripts (cont.)

- You should annotate your scripts with comments.

  ```
  # This is a comment.
  ```

  What makes a good comment?

- Save your scripts with the `.R` file extension

  e.g., `1-Data-prep.R`.

- Install packages once, load them every time you use them. Place at the top of your script.

  ```
  library(tidyverse)
  library(here)
  ```

Explore at the console; save anything that matters in a script.

Organising your scripts

You can organise your scripts with sections and headers.

```
# Section One ----------------------------

# Section Two ============================

### Section Three ########################
```

Organising your scripts

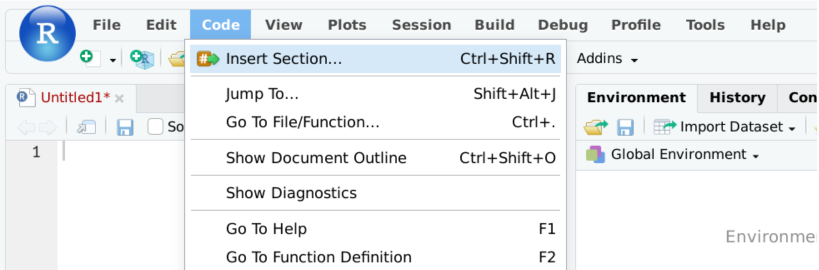
You can organise your scripts with sections and headers.

```
# Section One ---------------------------

# Section Two ===========================

#### Section Three #############################
```

These can be inserted in RStudio with Ctrl + ⇧ + R or Code › Insert Section...



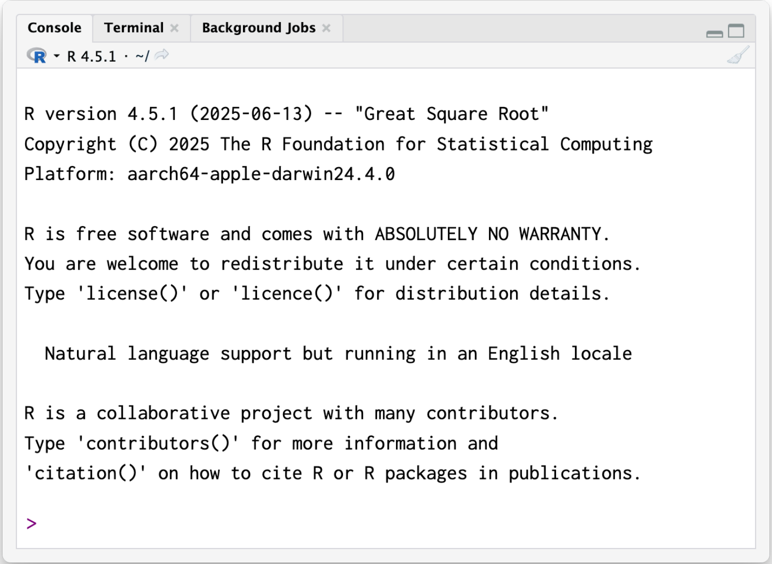
Combining several scripts

- Most analyses won't fit in a single script/file.
  - ⇝ You might have separate scripts for data import, cleaning, analysis, plotting...

- We can use the `source` command to execute code contained in another file.

```r
source("1-Import-raw-data.R")
source("2-Data-cleaning.R")
```

The R console



```
R version 4.5.1 (2025-06-13) -- "Great Square Root"
Copyright (C) 2025 The R Foundation for Statistical Computing
Platform: aarch64-apple-darwin24.4.0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

>
```

Tips for the R console

- Access previous commands with ↑ / ↓.
- Search previous commands with Ctrl + R.
- Clear the screen with Ctrl + L.

Operators

> *Symbols that tell the compiler to perform specific mathematical or logical manipulations.*

Arithmetic

`+`	addition
`-`	subtraction
`*`	multiplication
`/`	division
`^` or `**`	raising to a power

Relational

`x < y`	less than
`x > y`	greater than
`x <= y`	less than or equal to
`x == y`	equal
`x != y`	not equal

For more, see `https://www.statmethods.net/management/operators.html`

Getting help

You can get help for any function with `?`.

```
?rnorm   # Help for the 'rnorm' function
?install.packages
```

All help files have a consistent structure:

Description → *Usage* → *Arguments* → *Details* → *See Also* → *Examples*

- I find the ‘Arguments’ and ‘Examples’ sections the most useful.
- It’s worth familiarising yourself with this structure.

Resources

1. **A Succinct Introduction to R**

   Steve Haroz

   `http://r-guide.steveharoz.com`

2. **R for Data Science**

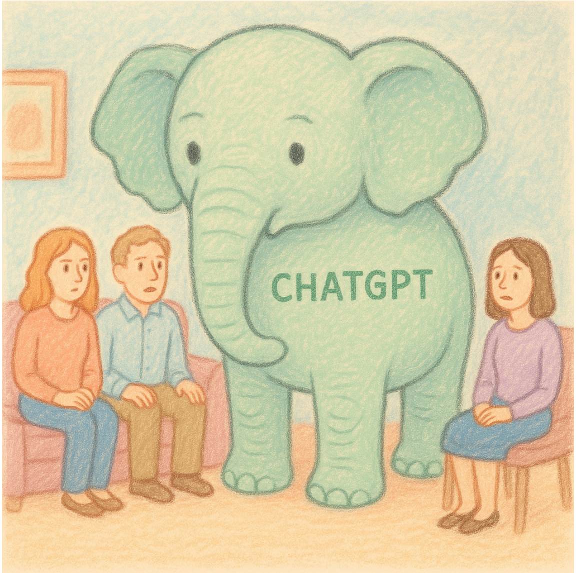
   Hadley Wickham and Garrett Grolemund

   `https://r4ds.hadley.nz`

3. **Advanced R**

   Hadley Wickham

   `https://adv-r.hadley.nz`



Should I use generative AI to learn R?

Should I use generative AI to learn R?

Yes:

- You need quick help debugging.
- You want to understand why your code isn't working.
- You know what you're doing, but want to work faster.

No:

- You're facing a complex problem you don't yet understand.
- Convoluted, out-of-date solutions.
- Data science has minimal boiler plate code.
- We need more thinking not less.