HKU SPACE

Msc Biomedical Science

MODULE 3: PROGRAMMING FOR BIOLOGICAL DATA

Session 1: The R Environment & Statistical Forensics

GU Haogao

The University of Hong Kong

guhaogao@hku.hk

2025-01-05

About me

www.guhaogao.com

- Research Assistant Professor in School of Public Health, HKU
- My research focuses on viral evolution (SARS-CoV-2, Influenza) and molecular epidemiology. My daily work involves transforming complex biological questions into data-driven answers.
- Skills: NGS data processing, public data collection, model building/fitting, debugging, high-performance computing, theoretical biology, etc..

The Learning Objective

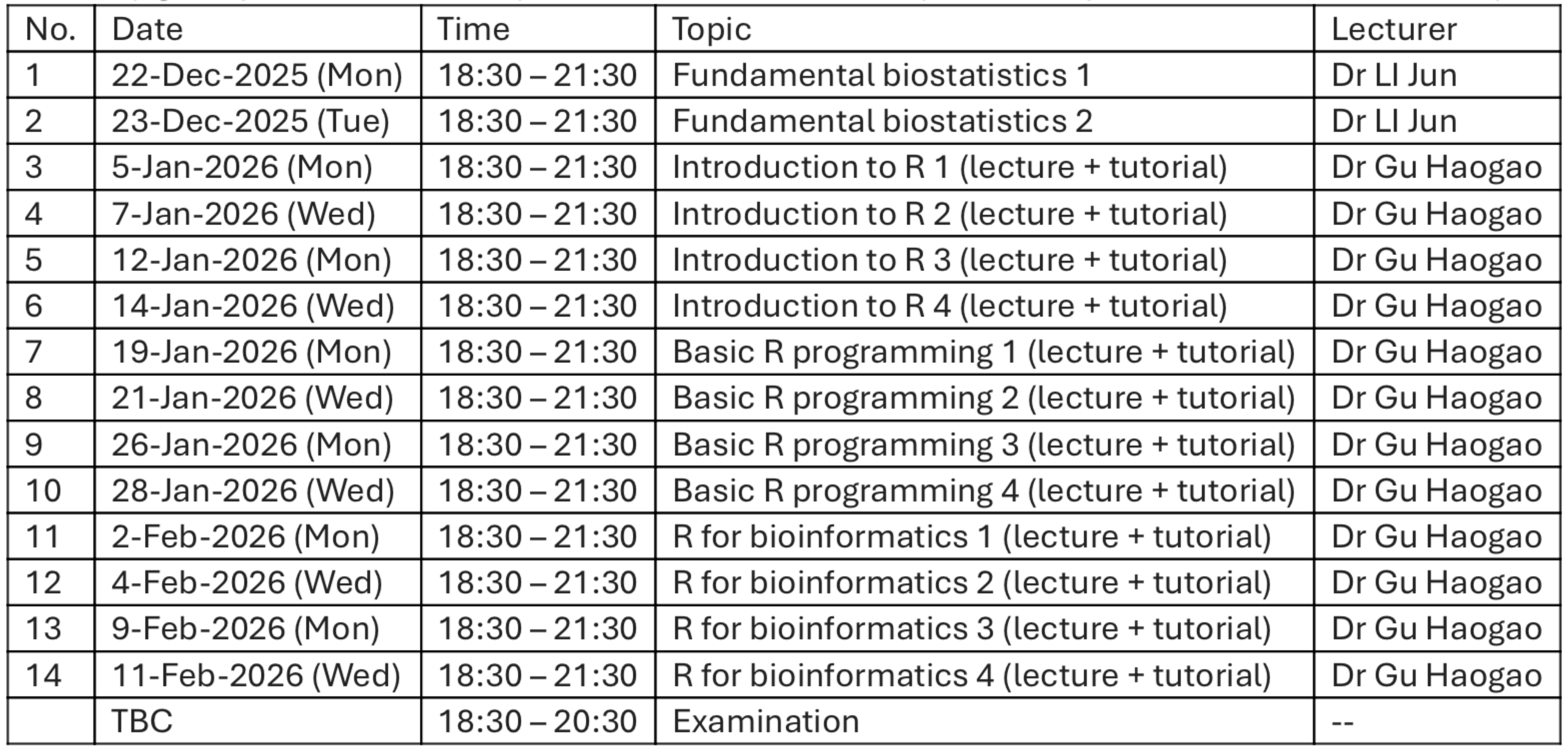
PROGRAMMING FOR BIOLOGICAL DATA

- This module aims to reinforce students' knowledge in biostatistics.
- It also introduces fundamental R programming and its application in analysing biological data to those students with no previous programming background.
- On completion of the module, students should be able to

1. discuss the key concepts in statistics and apply them in analysis of biological data;
2. discuss and apply the key concepts in programming;
3. develop small size, well-structured scripts with R for fundamental tasks;
4. process sequencing data that arise in bioinformatics.

Rundown

- Goal for this module: To guide you from manual "wet lab" logic to automated "dry lab" protocols.
- Teaching schedule:
  - Lecture (40 mins, 18:30 - 19:10)
  - Break (10 mins, 19:10 - 19:20)
  - Lecture (40 mins, 19:20 - 20:00)
  - Break (10 mins, 20:00 - 20:10)
  - Tutorial (80 mins, 20:10 - 21:30)

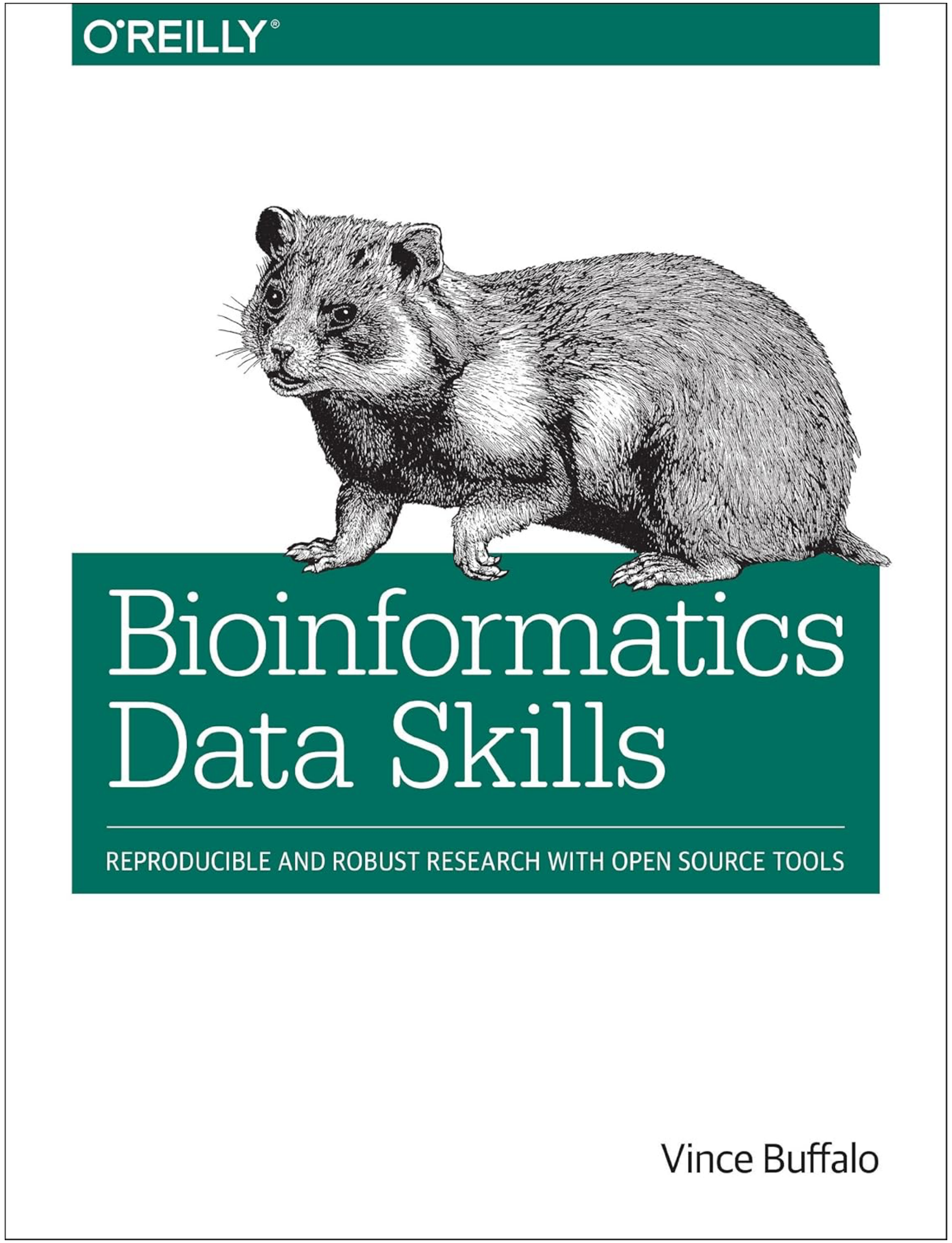


No.	Date	Time	Topic	Lecturer
1	22-Dec-2025 (Mon)	18:30 – 21:30	Fundamental biostatistics 1	Dr LI Jun
2	23-Dec-2025 (Tue)	18:30 – 21:30	Fundamental biostatistics 2	Dr LI Jun
3	5-Jan-2026 (Mon)	18:30 – 21:30	Introduction to R 1 (lecture + tutorial)	Dr Gu Haogao
4	7-Jan-2026 (Wed)	18:30 – 21:30	Introduction to R 2 (lecture + tutorial)	Dr Gu Haogao
5	12-Jan-2026 (Mon)	18:30 – 21:30	Introduction to R 3 (lecture + tutorial)	Dr Gu Haogao
6	14-Jan-2026 (Wed)	18:30 – 21:30	Introduction to R 4 (lecture + tutorial)	Dr Gu Haogao
7	19-Jan-2026 (Mon)	18:30 – 21:30	Basic R programming 1 (lecture + tutorial)	Dr Gu Haogao
8	21-Jan-2026 (Wed)	18:30 – 21:30	Basic R programming 2 (lecture + tutorial)	Dr Gu Haogao
9	26-Jan-2026 (Mon)	18:30 – 21:30	Basic R programming 3 (lecture + tutorial)	Dr Gu Haogao
10	28-Jan-2026 (Wed)	18:30 – 21:30	Basic R programming 4 (lecture + tutorial)	Dr Gu Haogao
11	2-Feb-2026 (Mon)	18:30 – 21:30	R for bioinformatics 1 (lecture + tutorial)	Dr Gu Haogao
12	4-Feb-2026 (Wed)	18:30 – 21:30	R for bioinformatics 2 (lecture + tutorial)	Dr Gu Haogao
13	9-Feb-2026 (Mon)	18:30 – 21:30	R for bioinformatics 3 (lecture + tutorial)	Dr Gu Haogao
14	11-Feb-2026 (Wed)	18:30 – 21:30	R for bioinformatics 4 (lecture + tutorial)	Dr Gu Haogao
	TBC	18:30 – 20:30	Examination	--

Acknowledgment

Reference book

- Bioinformatics Data Skills by Vince Buffalo
- My favourite introductory book to Bioinformatics
  - Useful
  - Funny
  - Easy to follow



O'REILLY

Bioinformatics Data Skills

REPRODUCIBLE AND ROBUST RESEARCH WITH OPEN SOURCE TOOLS

Vince Buffalo

Reproducible and Robust Research

Why programming?

- Increasing complexity of bioinformatic analyses makes findings error-prone.
  - To make things worse, some analyses are usually only run once before publication.
  - Some analyses may rely on very specific versions of all software/systems used.

- Healthy skepticism: never trust your tools (or data).
- Keeping the analyses reproducible and robust.

Reproducible and Robust Research

Why programming?

- Every conclusion/claim should be **reproducible**:
  - Karl Popper, *The Logic of Scientific Discovery*: "*non-reproducible single occurrences are of no significance to science*" (1959)
- Adopting Robust and Reproducible Practices Will **Make Your Life Easier**, Too:
  - You will almost certainly have to **rerun** an analysis more than once, possibly with new or changed data.
  - In the future, you will almost certainly **forget** the details. Without writing down key facts (e.g., where you downloaded data from, when you downloaded it, and what steps you ran), you'll certainly forget them. Documenting your computational work is equivalent to keeping a detailed lab notebook—an absolutely crucial part of science.

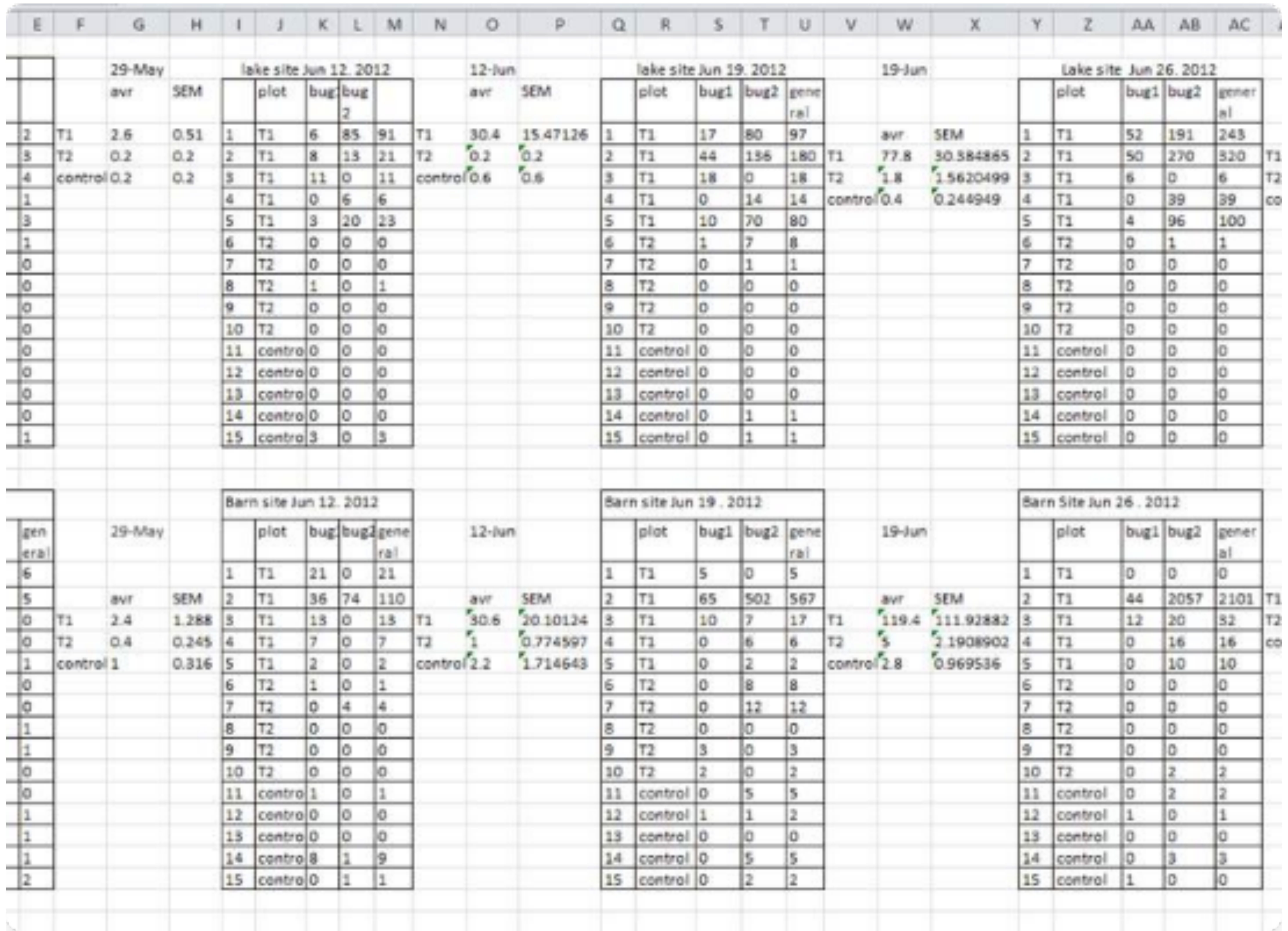
Reproducible and Robust Research

Recommendation for reproducible researches

- **Release** your **code and data**.
- **Document everything**.
- Make figures and statistics the **results of scripts**.
- Use **code** as documentation.
- Check **data integrity**:
    - MD5 is a method that turns any piece of data into a unique, fixed-size "fingerprint" or hash, used to verify data integrity.
    - 2024-12-31: a448e7af4416d615121f68d83c954ec5
    - 51fcad48f76a22206b5b5e1dd5df4507 (Guess this one)

The "Excel Trap" in Diagnostics

- Fragility: Data and calculations are mixed in the same cells. A single accidental keystroke can delete critical patient results forever.
- Lack of Audit Trail: "How did we get this CV%?" If the logic isn't recorded, the result cannot be verified.
- Manual Error: Dragging and dropping cells is prone to "off-by-one" errors that are hard to detect in large datasets.



E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC
		29-May		lake site Jun 12. 2012						12-Jun			lake site Jun 19. 2012					19-Jun			Lake site Jun 26. 2012			
		avr	SEM		plot	bug	bug 2			avr	SEM		plot	bug1	bug2	gene ral					plot	bug1	bug2	gener al
2	T1	2.6	0.51	1	T1	6	85	91	T1	30.4	15.47126	1	T1	17	80	97		avr	SEM	1	T1	52	191	243
3	T2	0.2	0.2	2	T1	8	13	21	T2	0.2	0.2	2	T1	44	136	180	T1	77.8	30.384865	2	T1	50	270	320
4	control	0.2	0.2	3	T1	11	0	11	control	0.6	0.6	3	T1	18	0	18	T2	1.8	1.5620499	3	T1	6	0	6
1				4	T1	0	6	6				4	T1	0	14	14	control	0.4	0.244949	4	T1	0	39	39
3				5	T1	3	20	23				5	T1	10	70	80				5	T1	4	96	100
1				6	T2	0	0	0				6	T2	1	7	8				6	T2	0	1	1
0				7	T2	0	0	0				7	T2	0	1	1				7	T2	0	0	0
0				8	T2	1	0	1				8	T2	0	0	0				8	T2	0	0	0
0				9	T2	0	0	0				9	T2	0	0	0				9	T2	0	0	0
0				10	T2	0	0	0				10	T2	0	0	0				10	T2	0	0	0
0				11	contro	0	0	0				11	control	0	0	0				11	control	0	0	0
0				12	contro	0	0	0				12	control	0	0	0				12	control	0	0	0
0				13	contro	0	0	0				13	control	0	0	0				13	control	0	0	0
0				14	contro	0	0	0				14	control	0	1	1				14	control	0	0	0
1				15	contro	3	0	3				15	control	0	1	1				15	control	0	0	0
				Barn site Jun 12. 2012								Barn site Jun 19 . 2012								Barn Site Jun 26 . 2012				
gen eral		29-May			plot	bug	bug2	gene ral		12-Jun			plot	bug1	bug2	gene ral		19-Jun			plot	bug1	bug2	gener al
6				1	T1	21	0	21				1	T1	5	0	5				1	T1	0	0	0
5		avr	SEM	2	T1	36	74	110		avr	SEM	2	T1	65	502	567		avr	SEM	2	T1	44	2057	2101
0	T1	2.4	1.288	3	T1	13	0	13	T1	30.6	20.10124	3	T1	10	7	17	T1	119.4	111.92882	3	T1	12	20	32
0	T2	0.4	0.245	4	T1	7	0	7	T2	1	0.774597	4	T1	0	6	6	T2	5	2.1908902	4	T1	0	16	16
1	control	1	0.316	5	T1	2	0	2	control	2.2	1.714643	5	T1	0	2	2	control	2.8	0.969536	5	T1	0	10	10
0				6	T2	1	0	1				6	T2	0	8	8				6	T2	0	0	0
0				7	T2	0	4	4				7	T2	0	12	12				7	T2	0	0	0
1				8	T2	0	0	0				8	T2	0	0	0				8	T2	0	0	0
1				9	T2	0	0	0				9	T2	3	0	3				9	T2	0	0	0
0				10	T2	0	0	0				10	T2	2	0	2				10	T2	0	2	2
0				11	contro	1	0	1				11	control	0	5	5				11	control	0	2	2
1				12	contro	0	0	0				12	control	1	1	2				12	control	1	0	1
1				13	contro	0	0	0				13	control	0	0	0				13	control	0	0	0
1				14	contro	8	1	9				14	control	0	5	5				14	control	0	3	3
2				15	contro	0	1	1				15	control	0	2	2				15	control	1	0	0

The Solution: "Code as Protocol"

The Wet Lab

You follow an **SOP** (Standard Operating Procedure) document.

You program the **thermocycler** with specific steps.

The machine executes the physical protocol.

The Dry Lab (R)

You write a **Script** (your digital SOP).

You feed the script to the **R Environment**.

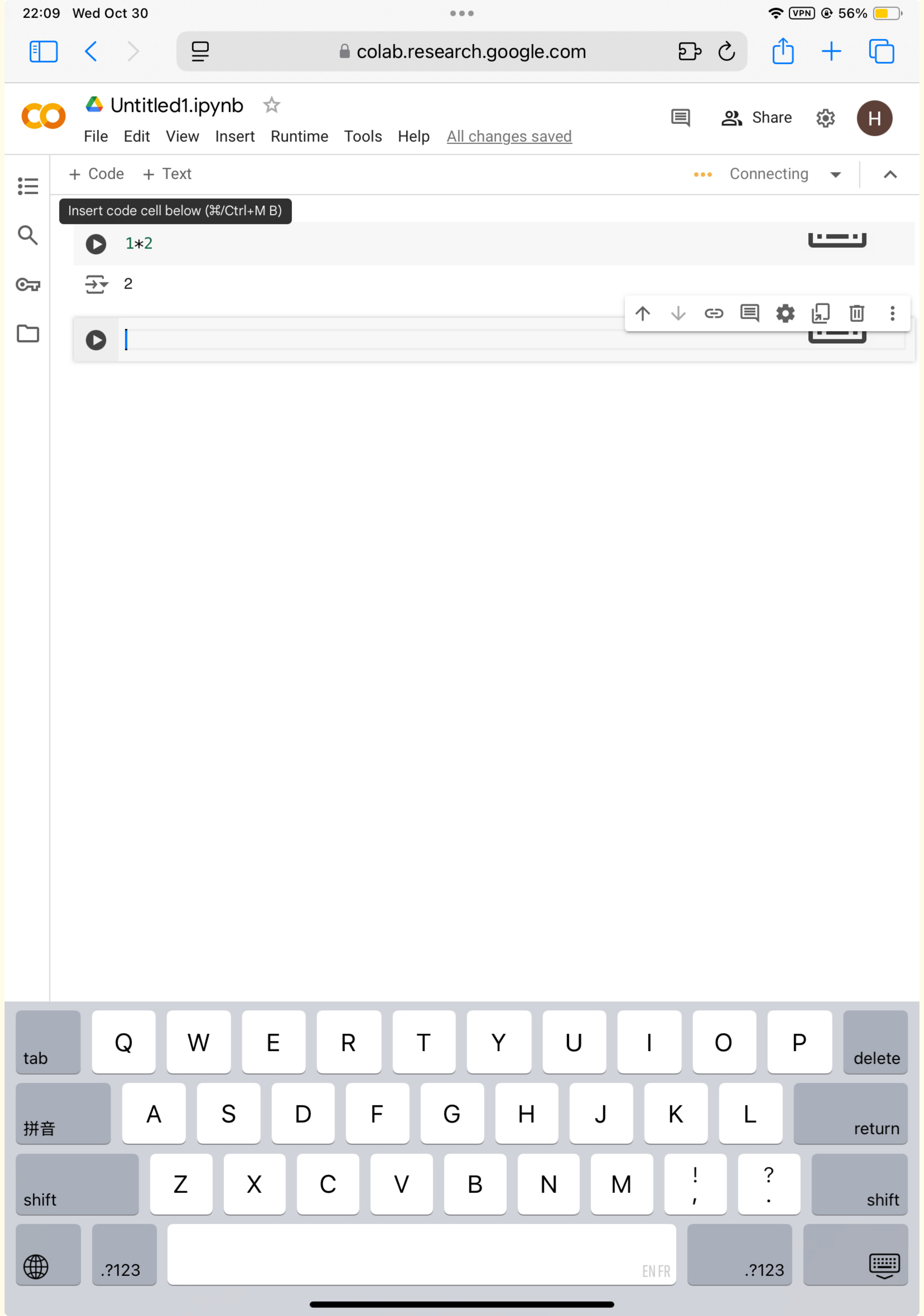
The computer executes the analysis protocol.

Reproducible every time.

Your Digital Bench: Google Colab

We use Google Colab to bypass hospital IT restrictions and ensure a standardized environment.

- **Cloud-Based:** No installation required. Runs entirely in your browser.
- **"Fresh Bench":** Every time you open it, it is a clean environment. It resets when you leave, ensuring no "contamination" from previous sessions.
- **Notebook Format:** Allows you to write narrative text (SOP) alongside your executable code.



```
1*2
```

```
2
```

Concept 1: Variables are "Test Tubes"

In the lab, you never hold liquid in your hands; you pour it into a labeled container. In R, a **Variable** is that container.

The Container

A named space to store your data value.

`patient_glucose`

The "Pour" Operator

The assignment operator <- puts the value into the variable.

`<-`

The Value

The actual data point you are storing.

`5.5`

```
patient_glucose <- 5.5
```

Concept 2: Vectors are "Batch Runs"

Rarely do we run just one sample in diagnostics. We run a batch. In R, we use the c() function to **Combine** multiple values into a single vector.

```
qc_level1 <- c(5.5, 5.6, 5.4, 5.5)
```

Think of this as filling a row in a tube rack or a 96-well plate. The vector holds the entire batch as one object.

Concept 3: Vectorization (Multi-Channel Pipette)

Efficiency in Action

This is the "magic" of R. If I want to convert units for 100 samples, I don't do it one by one.

I multiply the whole vector by the conversion factor. R applies the math to every element simultaneously.

```
qc_mmol <- qc_level1 * 0.0555
```

Just like using a multi-channel pipette to add reagent to every well at once.

Concept 4: Functions are "Instruments"

You don't need to know the internal electronics of a chemistry analyzer to use it. Similarly, you just need to know which R function is the right instrument for your data.

Input

Your raw data (Vector)

`qc_level1`

Instrument

The Function (Black Box)

`mean(...)`

Output

The Result

`5.5`

Common Instruments: mean(), sd(), round()

Concept 5: Data Types are "Sample Types"

You wouldn't put a tissue biopsy into a hematology analyzer. In R, you cannot mix different data types in one vector.

Numeric

Continuous data: Ct values, OD, Concentrations.

`5.5, 12.1`

Character

Text data: Patient IDs, Gender. **Must use quotes!**

`"Positive", "F"`

☑ Logical

Binary status: True or False statements.

`TRUE, FALSE`

Your R Toolkit for Today

- <- : **Assign** (Pour value into variable)
- c() : **Combine** (Create a vector/batch)
- mean() : **Instrument** to calculate Average
- sd() : **Instrument** to calculate Standard Deviation
- # : **Comment** (Notes to self, ignored by R)

Tutorial 1: The Daily QC Run

Goal: Manually generate a dataset, perform statistical forensics, and validate against manufacturer limits.

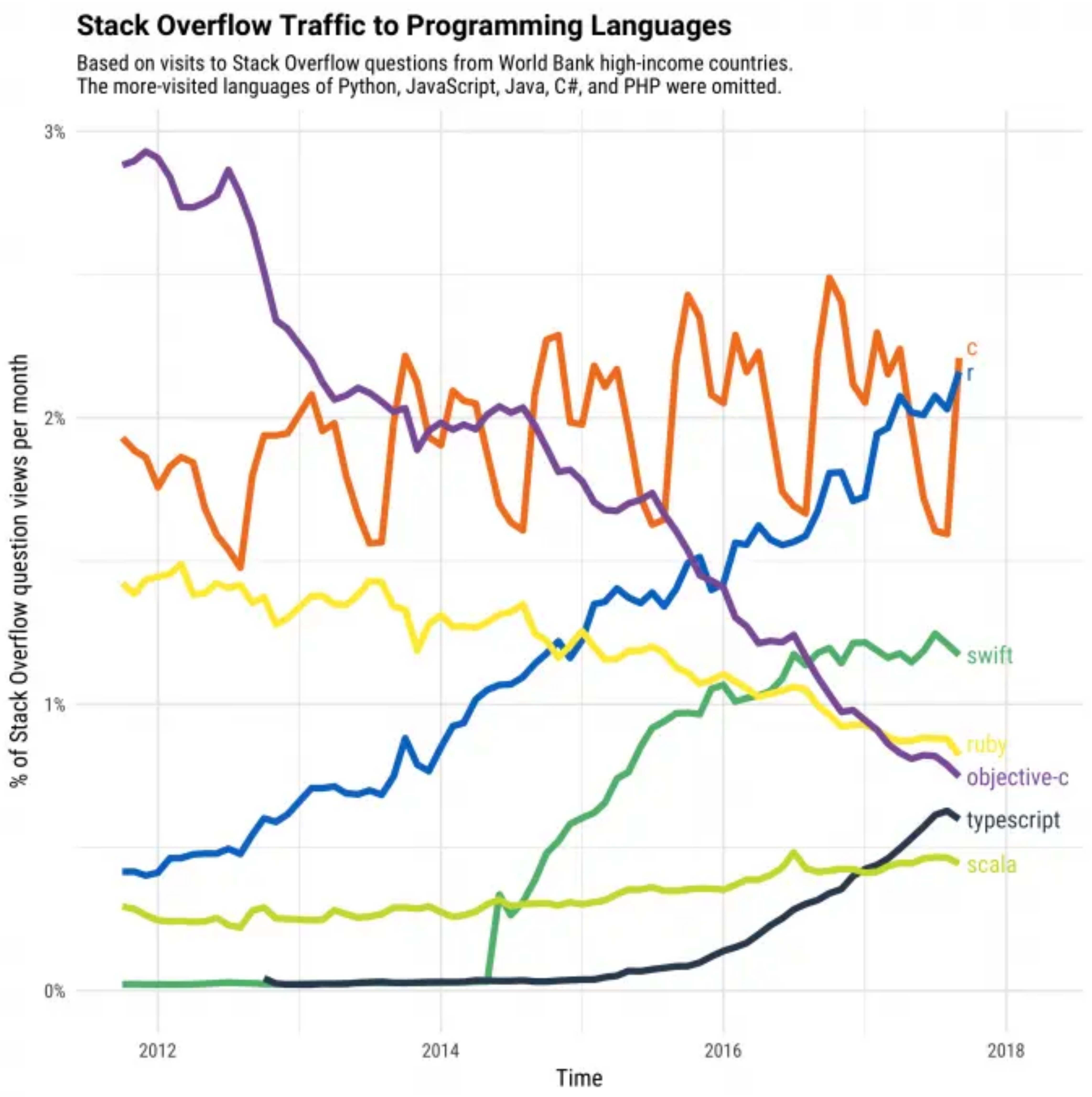
1. Go to the GitHub Link
2. Click "Open in Colab"
3. Run the Setup Chunk

A rapid introduction to the R language

R

Data analysis and visualisation

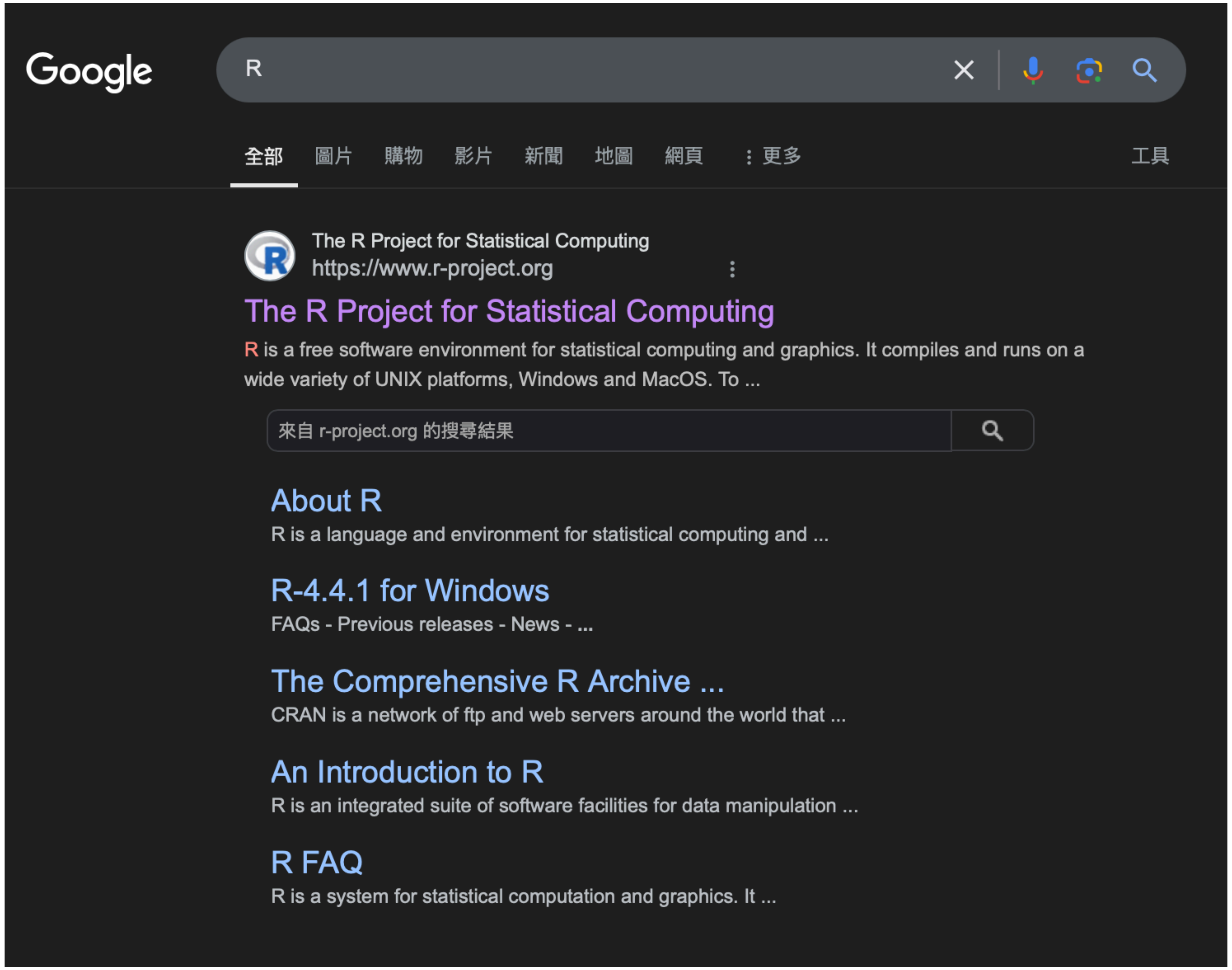
- How do you know R?
- Many bioinformaticians use R.
- Pros:
  - Easy!
  - Many existing packages.
- Cons:
  - Slow (relatively)



Running R

On your local machine

- To install R, go to the official website.
- Optionally you would also like to install the RStudio (an integrated development environment for R).
- Or you can use VSCode (like me).



The R Project for Statistical Computing
https://www.r-project.org

The R Project for Statistical Computing

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To ...

來自 r-project.org 的搜尋結果

About R
R is a language and environment for statistical computing and ...

R-4.4.1 for Windows
FAQs - Previous releases - News - ...

The Comprehensive R Archive ...
CRAN is a network of ftp and web servers around the world that ...

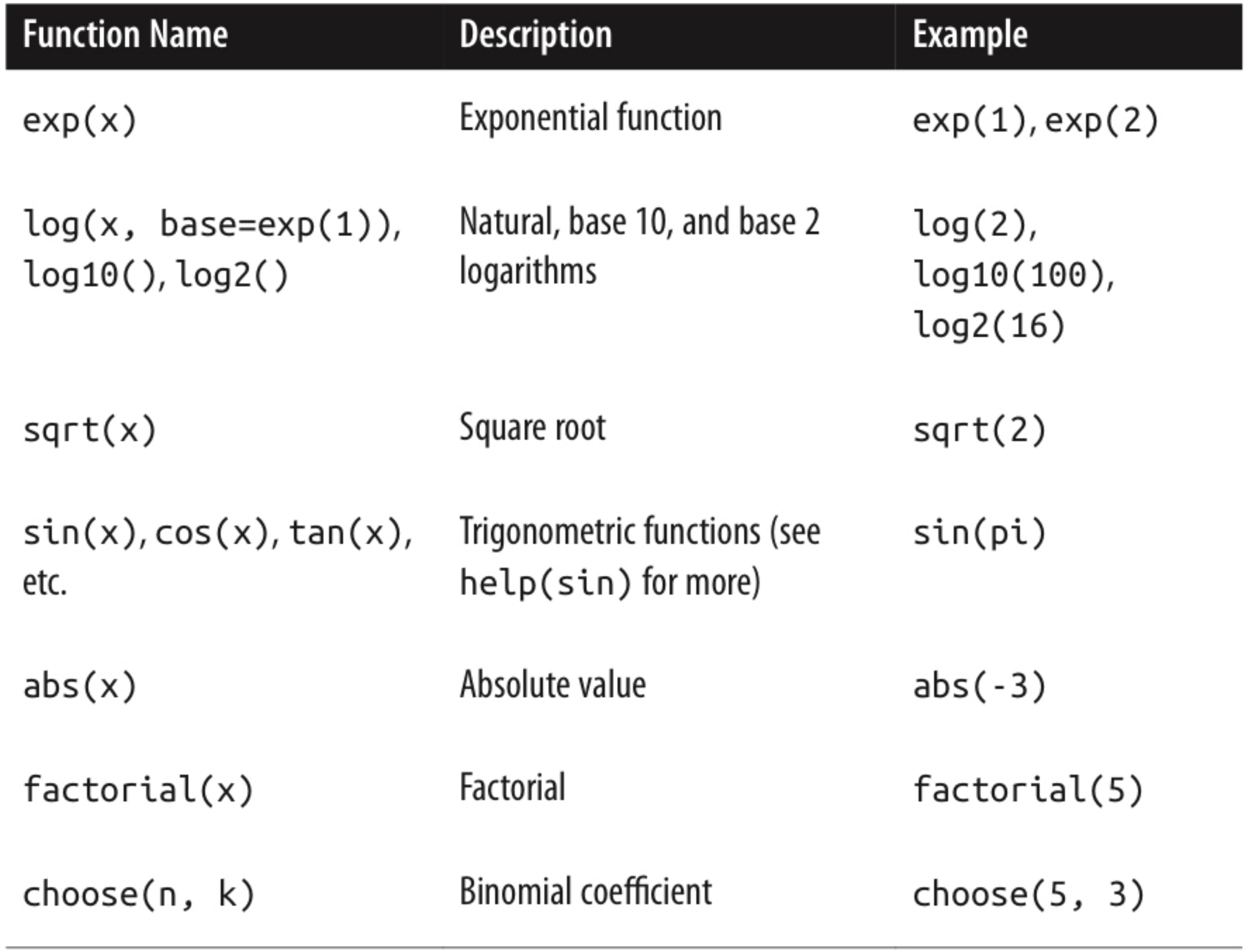
An Introduction to R
R is an integrated suite of software facilities for data manipulation ...

R FAQ
R is a system for statistical computation and graphics. It ...

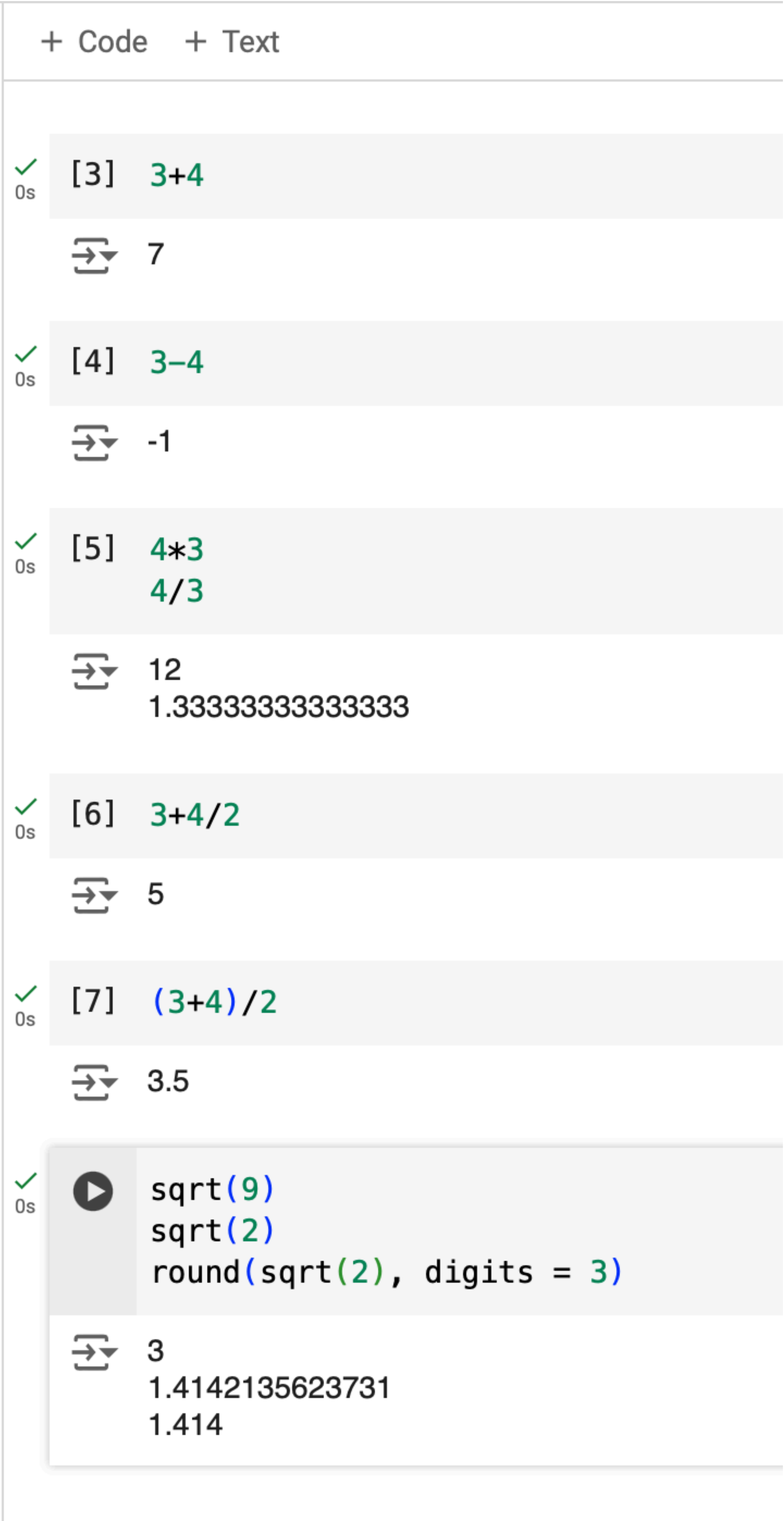
R language basics

Simple calculations

Table 8-1. Common mathematic functions



Function Name	Description	Example
`exp(x)`	Exponential function	`exp(1)`, `exp(2)`
`log(x, base=exp(1))`, `log10()`, `log2()`	Natural, base 10, and base 2 logarithms	`log(2)`, `log10(100)`, `log2(16)`
`sqrt(x)`	Square root	`sqrt(2)`
`sin(x)`, `cos(x)`, `tan(x)`, etc.	Trigonometric functions (see `help(sin)` for more)	`sin(pi)`
`abs(x)`	Absolute value	`abs(-3)`
`factorial(x)`	Factorial	`factorial(5)`
`choose(n, k)`	Binomial coefficient	`choose(5, 3)`



```
[3] 3+4
```

```
7
```

```
[4] 3-4
```

```
-1
```

```
[5] 4*3
    4/3
```

```
12
1.33333333333333
```

```
[6] 3+4/2
```

```
5
```

```
[7] (3+4)/2
```

```
3.5
```

```
sqrt(9)
sqrt(2)
round(sqrt(2), digits = 3)
```

```
3
1.4142135623731
1.414
```

Vectors, Vectorization, and Indexing

R’s feature

- Create a vector `x` containing three values (1, 9 and 36) using `c()`.
- Get the square root of `x` and store in `y`.
- Print x + y
- Print y
- Print the third component of y.
- Check whether the elements in y are greater than 4.
- Print the elements in y which is/are less than 5.

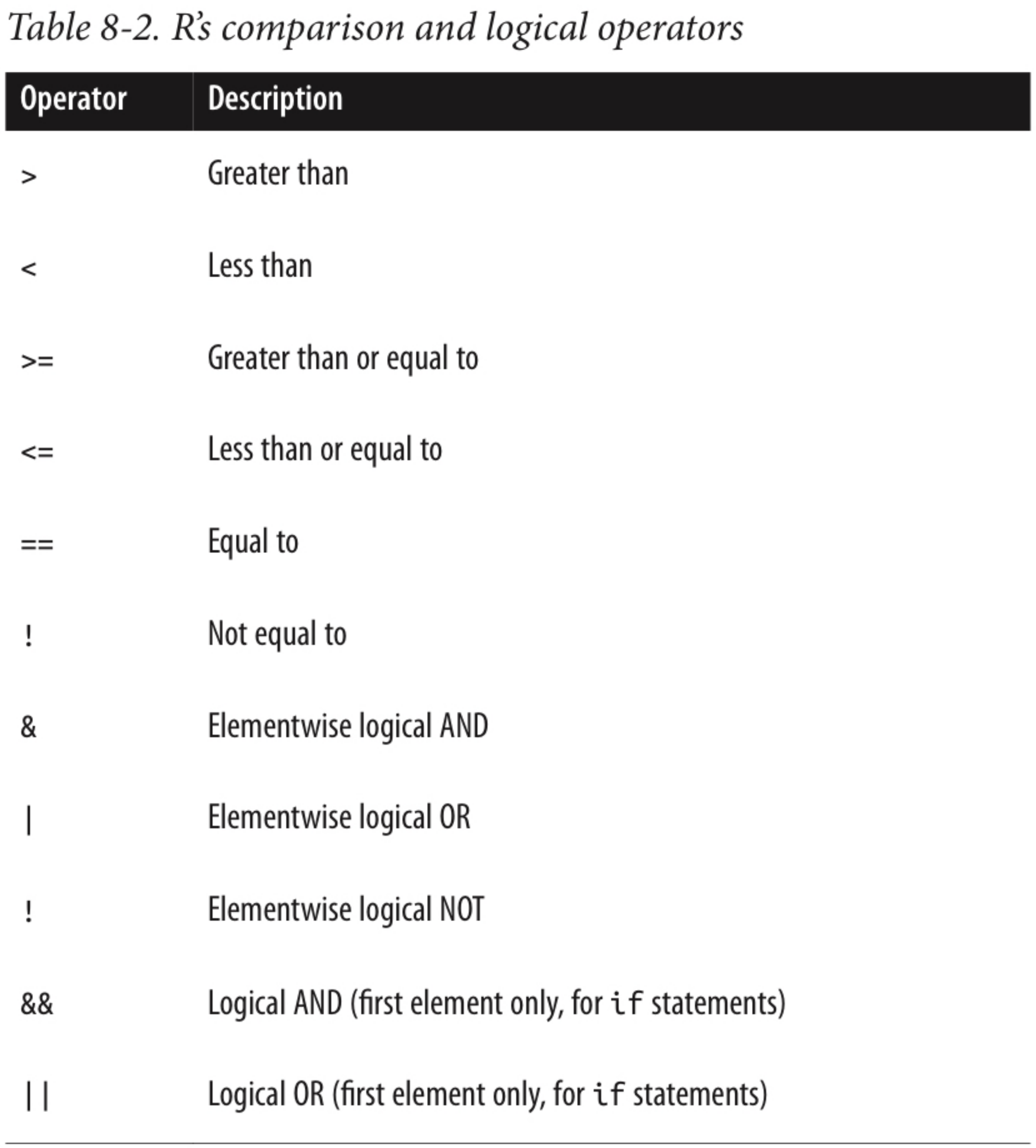
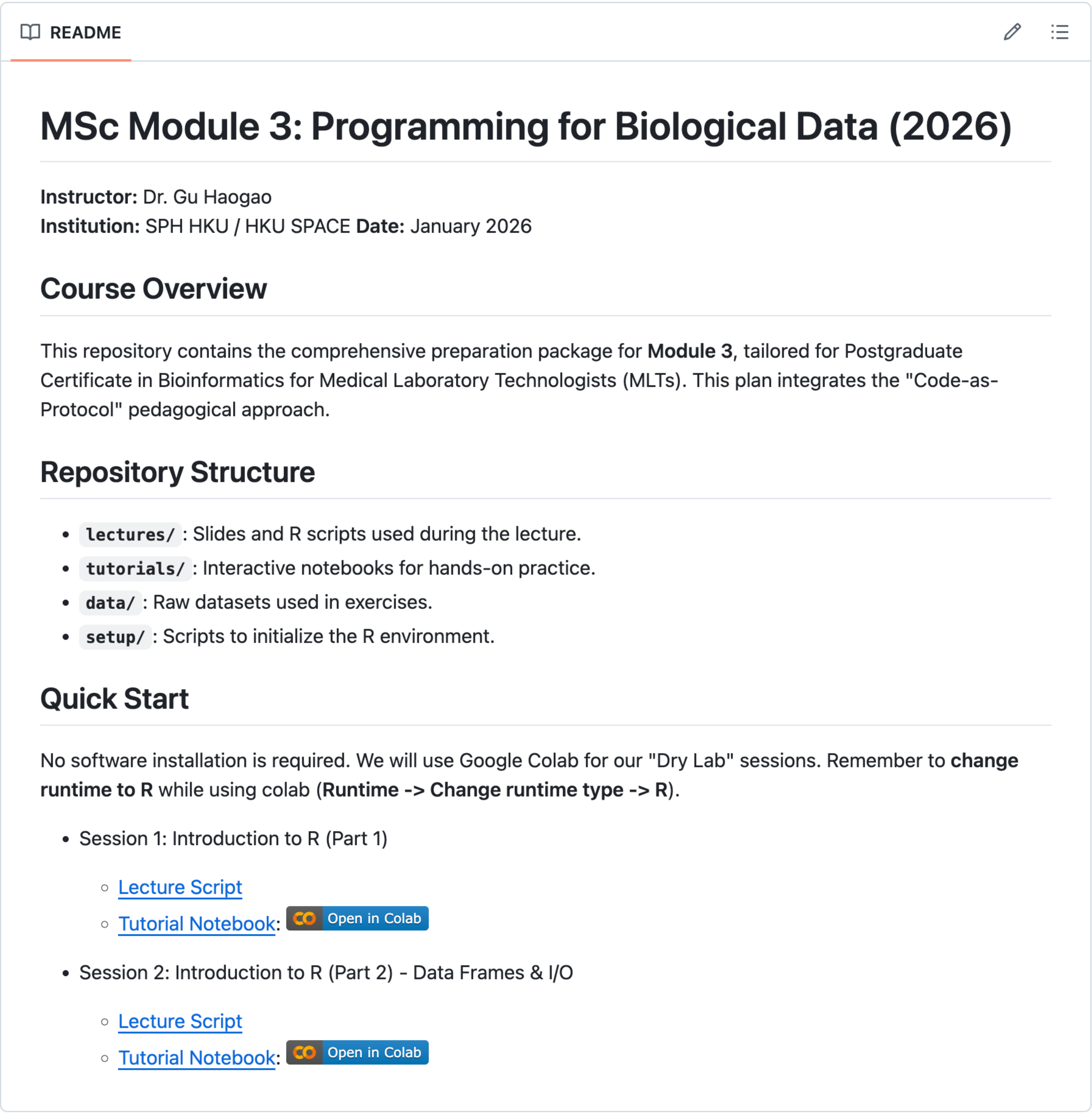


Table 8-2. R’s comparison and logical operators

Operator	Description
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
!	Not equal to
&	Elementwise logical AND
|	Elementwise logical OR
!	Elementwise logical NOT
&&	Logical AND (first element only, for `if` statements)
||	Logical OR (first element only, for `if` statements)

https://github.com/Koohoko/MSc_Module3_Programming_BioData_HKUSpacce_2026



MSc Module 3: Programming for Biological Data (2026)

Instructor: Dr. Gu Haogao
Institution: SPH HKU / HKU SPACE **Date:** January 2026

Course Overview

This repository contains the comprehensive preparation package for **Module 3**, tailored for Postgraduate Certificate in Bioinformatics for Medical Laboratory Technologists (MLTs). This plan integrates the "Code-as-Protocol" pedagogical approach.

Repository Structure

- `lectures/` : Slides and R scripts used during the lecture.
- `tutorials/` : Interactive notebooks for hands-on practice.
- `data/` : Raw datasets used in exercises.
- `setup/` : Scripts to initialize the R environment.

Quick Start

No software installation is required. We will use Google Colab for our "Dry Lab" sessions. Remember to **change runtime to R** while using colab (**Runtime -> Change runtime type -> R**).

- Session 1: Introduction to R (Part 1)
  - Lecture Script
  - Tutorial Notebook: Open in Colab
- Session 2: Introduction to R (Part 2) - Data Frames & I/O
  - Lecture Script
  - Tutorial Notebook: Open in Colab