ANALYSING GENOMIC AND TRANSCRIPTOMIC DATA IN R

A mini-course

Overview of the course

Date	1st Lecture	2nd Lecture
16th July	Icebreakers, ground rules and maybe and Intro to Molecular Biology and NGS	
17th July	Intro to Molecular Biology and NGS	Data analysis 1
18th July	Experiment: Nucleic acid extraction	Data analysis 2
19th July	Single-cell DNA, RNA and protein technologies	Proteomics, spatial technologies and epigenomics Data analysis 3
20th July	Data analysis 4	Experiment: Staining our own cells
21st July	Experiment: Gel electrophoresis	Data analysis 5 Data analysis 6
22nd July	Preparation for the final presentation	Final presentation and closing ceremony

Mini-course structure

- Session 1: R basics and data exploration
- Session 2: Data exploration and data visualisation

This covers up to chapter 2

- Session 4: Quality Check (QC) and Alignment
- Session 5: Statistics, Clustering and Dimensionality Reduction
- Session 6: RNA-seq analysis

Aim

- To be able to explore and visualise genomic data in R.

What is bioinformatics?

Combines biology, computer science, and statistics to analyse and interpret biological data.

Steps of genomic analysis

1. Data collection
2. Data quality check (QC) and cleaning
3. Data preprocessing
4. Data exploration
5. Data visualisation

SESSION 1

R SETUP

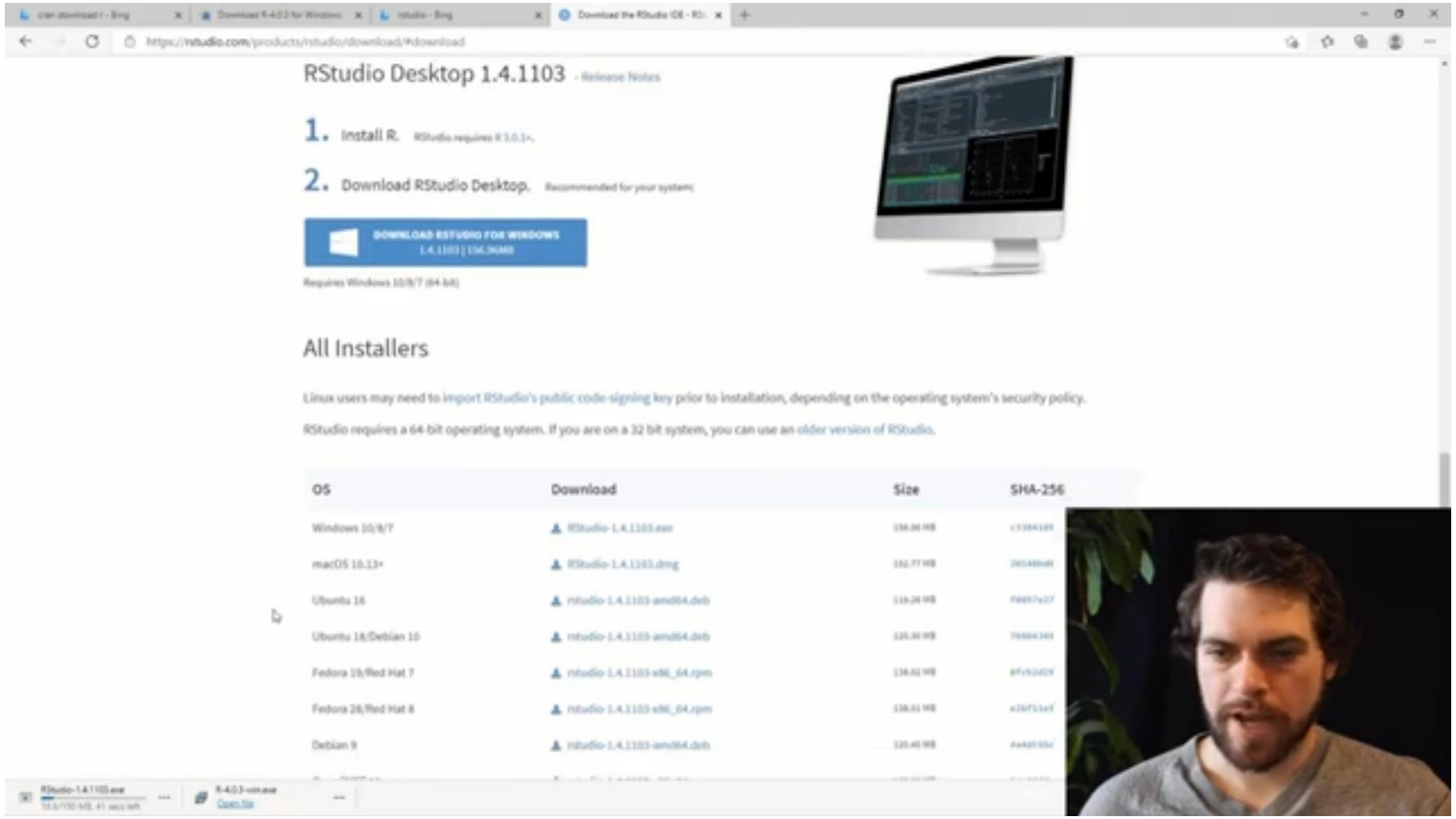
Download R and R Studio

https://posit.co/download/rstudio-desktop/

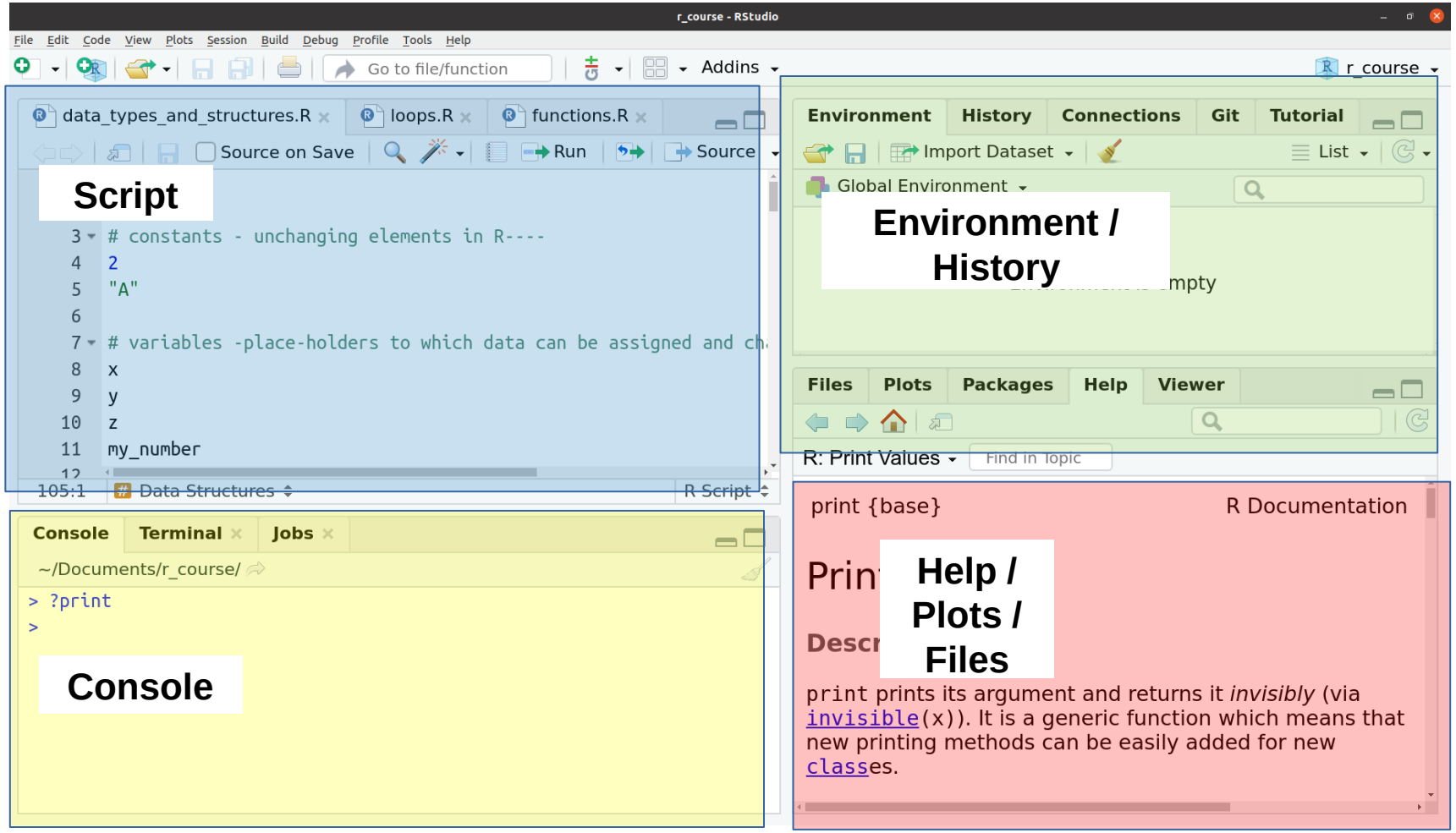
Download the latest versions of R and RStudio for your PC

- R is the underlying statistical computing environment
- RStudio is a graphical integrated development environment (IDE) that makes using R much easier and more interactive.

Download R and R Studio



R studio features



Download packages

- To install packages

install.packages(“tidyverse”)

- To use installed packages

library(tidyverse)

R BASICS

Objectives

1. Constants and variables
2. Data types and data structures
3. Reading and writing data
4. Basic data exploration

Help in R

- Type into R:

? And ??

Help()

- Stack Overflow for online help

https://stackoverflow.com/

What are constants and variables?

- Constants are fixed.
  - Pi
  - Avogadro's number
- Variables are assigned
  - X=5
  - Y=2

Basic commands

- Arithmetic operations: +, -, *, /, ^
- Assignment: <- or =]
- R comments: #

Try

```
a <- 5

b <- 10

c <- a + b

print(c)
```

Data types

- Numeric: 42, 3.14
- Character: "Hello, world!"
- Logical: TRUE, FALSE
- Factor: ("apple", "banana", "apple")

Try

```
num <- 42
char <- "Hello, world!"
bool <- TRUE
fact_var <- factor(c("apple", "banana", "apple"))
```

Type of functions

- Functions: A set of instructions that perform a specific task
- Built-in functions: Pre-defined and available in standard R packages
  - In R, built in functions have ()
  - For example, merge()
- User-defined functions: created by the user to perform specific tasks not covered by built-in functions.

Conversion between data types

- as.character()
- as.numeric()
- as.logical()
- as.factor(c())

Data structures

- Vectors
- Matrices
- Lists
- Data Frames

Vectors

- One-dimensional arrays.
- Created using c()

Try

vec <- c(1, 2, 3, 4, 5)

days <- c("Monday", "Tuesday", "Wednesday", "Thursday"," Friday", "Saturday" and "Sunday")

Matrices

- Two-dimensional arrays with elements of the same type.
- Created using matrix()

Try

```
mat <- matrix(1:9, nrow = 3, ncol = 3)
```

Lists

- Ordered collections of objects, which can be of different types.
- Created using list()

Try

```
lst <- list(num = 42, char = "Hello", bool = TRUE)
```

How about typing ‘number’ instead of ’num’?

Data frames

- Two-dimensional tables with columns of different types.
- Created using data.frame().

Try

```
df <- data.frame(
  x = c(1, 2, 3),
  y = c("A", "B", "C")
)
```

Sub-setting data structures

- vec[1] # First element of the vector
- mat[2, 3] # Element in the 2nd row, 3rd column of the matrix
- lst$name # Accessing the 'name' element from the list
- df$Name # Accessing the 'Name' column from the data frame

Converting between data structures

- as.list(df) # Converting an object to list
- as.data.frame(mat or list or characyter) # Converting matrix, list or character to data frame

Exercise 1

Objectives

✓Constants and variables

✓Data types and data structures

3. Reading and writing data
4. Basic data exploration

Reading and writing data

- read.table()
- read.csv()
- write.table()
- write.csv()

Excel

- Package needed: readxl
- library(readxl)

read_excel()

read_xls()

read_xlsx()

Setting working directory

- getwd() # Get current working directory
- setwd("/path/to/your/directory") # Set working directory
- list.files() # List files in the working directory

How to view data?

- data() #load specified datasets that are built into R packages into the R environment.
- head() # View the first 6 rows of the data
- tail() # View the last 6 rows of the data
- str() # Get the structure of the data
- summary() # Get summary statistics for the data

Cleaning Data

- What is data cleaning: preparing raw data for analysis
- Why clean data:
  - Handling missing data
  - Removing unnecessary rows/columns
  - Standardising formats
  - Removing duplicates
  - Correcting errors

```
na.omit() # Remove rows with missing values

df[complete.cases(df), ] # Another way to remove rows with missing values

df[, -5] # Remove the 5th column

df[-10, ] # Remove the 10th row
```

Exercise 2

Objectives

- ✓Constants and variables
- ✓Data types and data structures
- ✓Reading and writing data
- ✓Basic data exploration

Any questions?

R mini-project