R Statistical computing

START HERE

`R`
Start interactive session

`summary(df)`
Quick statistics

`x ← c(1, 2, 3)`
Make a vector

`df[df$`*`<col>`*`> 5, ]`
Filter rows

`df ← read.csv("`*`<file>`*`")`
Load a CSV

`?`*`<topic>`*
Open help page

`str(df)`
Inspect structure

`Rscript` *`<file.R>`*
Run a script

Vector one type → `names + dims` → **List** mixed types → `equal-length columns` → **Data frame** table of columns → `model / plot` → **Result** print or write

CONCEPTS

Almost everything is a vector; a single number is a vector of length 1. Operations work on whole vectors at once.

Indexing starts at 1, and negative indices drop elements: x[-1] is everything but the first.

← assigns, == compares. = also assigns at top level but only names arguments inside calls.

RUNNING R

`R`
Start an interactive session

`Rscript` *`<file.R>`* **`[args]`**
Run a script non-interactively

`Rscript -e '`*`<expr>`***`'`**
Evaluate an expression and exit

`R --vanilla`
Ignore profile, site and saved state
Short for --no-save --no-restore --no-site-file --no-init-file --no-environ. Use it to rule out your own startup files.

`R --no-save`
Do not offer to save the workspace

`commandArgs(trailingOnly = TRUE)`
Script arguments as a character vector

`q()`
Quit the session

`source("`*`<file.R>`***`")`**
Run a script inside a session

HELP AND PACKAGES

`?`*`<topic>`***
Open the help page for a function

`??`*`<term>`***
Search all help pages

`example(`*`<fn>`***`)`**
Run the examples from a help page

`apropos("`*`<part>`***`")`**
List objects matching a name part

`install.packages("`*`<pkg>`***`")`**
Install from CRAN

`library(`*`<pkg>`***`)`**
Attach an installed package

HELP AND PACKAGES (cont.)

`<pkg>`**`::`*****`<fn>`***
Call a function without attaching

`update.packages()`
Update installed packages

VECTORS

`c(1, 2, 3)`
Combine values into a vector

`1:10`
Integer sequence

`seq(0, 1, by = 0.25)`
Sequence with a step

`seq_len(`*`<n>`***`)`**
1..n, safe when n is 0

`rep(`*`<x>`***`, times = 3)`**
Repeat values

`length(`*`<x>`***`)`**
Number of elements

`x[3]`
Third element

`x[c(1, 3)]`
Several elements by position

`x[x > 5]`
Elements matching a condition

`x[-1]`
Everything except the first

`which(x > 5)`
Positions where TRUE

`%in%`
Membership test: x %in% y

`sort(x) / order(x)`
Sorted values / sorting positions

`table(x)`
Count each value

`is.na(x)`
Test for missing values

`ifelse(`*`<test>`***`, `***`<yes>`***`, `***`<no>`***`)`**
Vectorised if/else

`paste("a", "b", sep = "-")`
Join strings

DATA STRUCTURES

`list(a = 1, b = "x")`
List of mixed types

`l$a / l[["a"]]`
Element of a list

`factor(x, levels = c("lo", "hi"))`
Categorical vector

`class(x) / typeof(x)`
Class / internal type

`str(x)`
Compact structure of any object

DATA STRUCTURES (cont.)

`as.numeric(x)`
Convert type (also as.character, as.integer)

DATA FRAMES

`data.frame(a = 1:3, b = c("x", "y", "z"))`
Build a data frame

`head(df, `*`<n>`***`)`**
First rows

`nrow(df) / ncol(df)`
Row / column count

`names(df)`
Column names

`summary(df)`
Per-column statistics

`df$`*`<col>`***
One column as a vector

`df[`*`<rows>`***`, `***`<cols>`***`]`**
Rows and columns by index or name

`df[df$`*`<col>`***` > 5, ]`**
Filter rows

`df[order(df$`*`<col>`***`), ]`**
Sort rows

`df$`*`<new>`*** **`← df$a * 2`**
Add a column

`subset(df, a > 5, select = c(a, b))`
Filter and pick columns

`merge(df1, df2, by = "`*`<key>`***`")`**
Join two frames on a key

`aggregate(a ~ g, data = df, FUN = mean)`
Group and summarise

CONTROL AND FUNCTIONS

`f ← function(x, y = 2) { x + y }`
Define a function

`if (`*`<cond>`***`) { } else { }`**
Conditional

`for (i in `*`<seq>`***`) { }`**
Loop over a sequence

`x |> f()`
Native pipe: f(x) (R 4.1+)

`tryCatch(`*`<expr>`***`, error = function(e) `***`<h>`***`)`**
Handle errors

`print(x) / cat("a", x, "\n")`
Print object / plain text

APPLY FAMILY

`sapply(x, f)`
Apply, simplify to vector or matrix

`lapply(x, f)`
Apply, always return a list

`apply(m, 1, sum)`
Over rows (2 for columns) of a matrix

`mean(x, na.rm = TRUE)`
Mean ignoring NA

`sd(x) / median(x) / quantile(x)`
Spread and location

IMPORT AND EXPORT

`read.csv("`*`<file>`***`")`**
Read a CSV into a data frame

`write.csv(df, "`*`<file>`***`", row.names = FALSE)`**
Write a data frame as CSV

`saveRDS(x, "`*`<file>`***`.rds")`**
Save one object

`readRDS("`*`<file>`***`.rds")`**
Load one object

`getwd() / setwd("`*`<dir>`***`")`**
Show / change working directory

PLOTTING

`plot(x, y)`
Scatter plot

`hist(x)`
Histogram

`plot(x, y, main = "`*`<t>`***`", xlab = "`***`<x>`***`", col = "red")`**
Title, label and colour

`pdf("`*`<file>`***`.pdf")`**
Send plots to a PDF; end with dev.off()

`dev.off()`
Close the current graphics device

RECIPES

Script with an argument

```
# run: Rscript top.R data.csv 5
args ← commandArgs(trailingOnly
= TRUE)
df ← read.csv(args[1])
print(head(df,
as.integer(args[2])))
```

Summarise by group

```
df ← read.csv("sales.csv")
res ← aggregate(amount ~
region, data = df, FUN = sum)
res ← res[order(-res$amount), ]
write.csv(res, "by_region.csv",
row.names = FALSE)
```