Scripting for data analysis

Martin Johnsson
IFM Biology

Content

Week 1 – today: A crash course in R

Homework 1: The unicorn dataset

Week 2 – February 1: Programming for data analysis

Homework 2: The unicorn expression dataset

Week 3 – February 13: Working with moderately large data

Homework 3: Design analysis by simulation

Content

Exercises – suggested solutions online

Homeworks – hand in to me before next seminar

Presentations:
https://people.ifm.liu.se/~marjon/scripting_for_data_analysis

GitHub repository:
https://github.com/mrtnj/scripting_for_data_analysis

1. A crash course in R

What?

free open source

statistical environment

scripting language

Why?

do statistics and graphs without having to worry about being locked out of the license server

access to particular packages (*limma* for gene expression, *R/qtl* for genetic mapping, *rstan* for Bayesian analysis etc)

automate repetitive tasks

... to write code!

Why scripting?

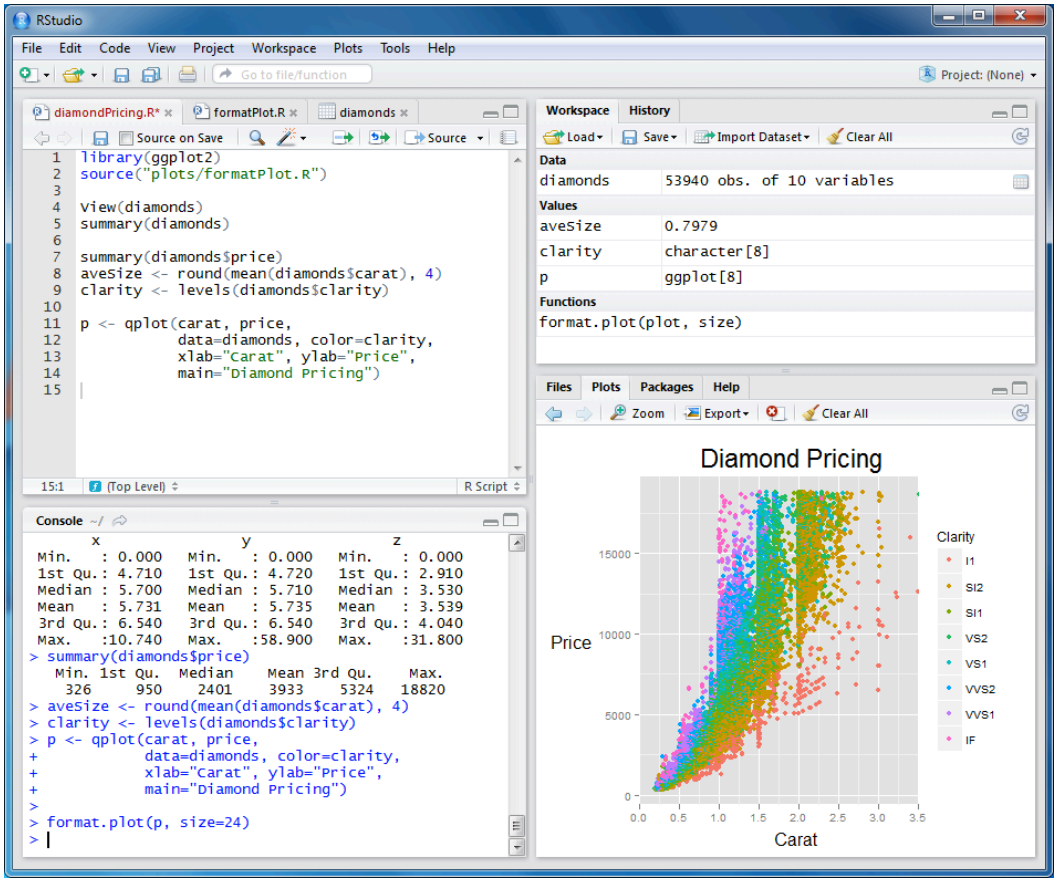
harder the first time

easier the next 20 times …

necessity for moderately large data

R-project.org

Rstudio.com



Alternatives

MATLAB – scientific computation, different set of packages

Python – scripting, scientific computation, bioinformatics

Perl – scripting, bioinformatics

Julia – scientific computation

C++, Java, etc – software development, scientific computation

Demo: RStudio, interactive use

Interface is immaterial

interfaces vary (ssh into a server, RStudio, alternatives)

language is universal

few platform-dependent elements that the end user needs to worry about

Help!

within R: ? and tab

your favourite search engine

asking (Stack Exchange, R-help mailing list, package mailing lists)

Scripting

write your code down in a .R file

run it with source ( )

comments

Reproducible analysis: make it run without intervention from start to finish

Coding style: write for humans (mostly: your future self), not computers

Concepts

Expression: `2 + 3 * 4 ^ 5`

Assignment: `my_variable <- 42`

Call a function: `some.function()`

Anatomy of a function call

```
function.name(parameter1 = 100,
              parameter2 = TRUE)
mean(some_data$column)
mean(x = some_data$column)
mean(x = some_data$column, na.rm = TRUE)
?mean
```

Reading in data

Excel sheet to data.frame

- one sheet at a time
- clear formatting
- short succinct column names
- export text

read.table, read.csv etc (base R)

read_csv etc (readr package, RStudio)

Import Dataset button

Demo: Read a csv file

1. Import Dataset > From CSV
2. Change settings until table looks good
3. Copy code into script

Working with columns

Columns are vectors.

columns in expressions

```
data$column1 + data$column2
log10(data$column2)
```

assignment arrow

```
data$new_column <- log10(data$column2)
```

Subsetting

Extract parts of a data frame based on contents.

logical operators ==, !=, >, <, !

```
subset(data, column1 == 1)
subset(data, column1 == 1 & column2 > 2)
```

Indexing with [ ]

Extract rows and columns of a data frame.

first three rows: `some_data[c(1,2,3), ]`

two columns: `some_data[, c(2,4)]`

ranges: `some_data[, 1:3]`

A package: ggplot2

The statistical graphics package we will use.

```
install.packages("ggplot2")
library(ggplot2)
qplot(x = one_variable,
      y = another_variable,
      data = some_data)
```

Demo: Scatterplot

```
library(ggplot2)

qplot(x = weight, y = horn.length, data = unicorns)

qplot(x = diet, y = horn.length,
      data = unciorns, geom = "jitter")

qplot(x = weight, y = horn.length,
      data = unicorns) +
  facet_grid(diet ~ colour)
```

Demo: Linear model

```
## fit the models
model <- lm(horn.length ~ diet + colour,
            data = unicorns)
model_int <- lm(horn.length ~ diet * colour,
                data = unicorns)
## extract coefficients
coefs <- coef(model)
## explore coefficients
coefs[1] + coefs[2] + coefs[3]
coefs[1]
## F-test
drop1(model, test = "F")
```

One last thing: Don’t save your workspace!

Exercise 1: data frames, linear models, coin toss example

Homework 1: the unicorn dataset